October 1990 £2.75

Commodore DISK·USER

RASTER BAR EDITOR

Colours galore for all and sundry

More Adventures in 'C'

HEXADECIMAL FOR BEGINNERS

A SYSOP TELLS ALL

9 770953 061007

NOW IS THE TIME
TO CATCH UP ON ISSUES YOU HAVE MISSED

VOL 3 No.6 APR '90
BAR PROMPTS - M/C input routine.
HI-LITE BARS - Basic inputs.
TEXAS DEMO - Basic in demos
CHARS TO SPRITES - Convert UDG's to sprites.
FONT FACTORY - Compliments program above.
3D TEXT MACHINE - 3D text screens the easy way.
SCREEN ENHANCER - Full screen use made easy
SPREADSHEET 64 - Easy to use spreadsheet
MINI-AID - 3 shorts to aid the Basic programmer.
C128 COLLECTION - 3 very useful C128 programs.

VOL 3 No.7 MAY '90
NUDGE - ELD explained.
WINDOW WIPER -alternative screen wipe system.
CHARACTER EXTRACTOR - Borrow those nice char sets.
MAZE GENERATOR - Create your own fun.
HIRES ANIMATOR - This difficult subject made easier.
SPRITE DRIVER - Platform game designing without the fuss.
ROTOTRON - Sight and sound.
TEXT COMPRESSION How to squeeze a gallon into a pint.
SCREENS - Make and save your own help screens.
INTERRUPT POINTERS - Geos style windows and pointers.

VOL 3 No.8 JUN '90
ALEATORY MUSIC - Alternative music.
SPRITE BASIC - Efficient sprite handling through Basic.
SPRITE GENERATOR - Another sprite editor.
MUNCHER - Pacman back with a vengeance.
ASTRODUS - Escape the spaceship in this adv.
1581 DIRECT ACCESS - Exploring the 1581.
PERSONAL ORGANISER - Design your own organiser pages.
128 CONVERTOR and MATHS AID - 2 more for C128 users.

VOL 3 No.9 JUL '90
QUICK MERGE 64/128 - Another useful routine. THE GAME PLAN - Whats where in games.
CHARACTER DESIGNER - Another char designer.
HASHBASE 128 - C128 Database.
REVASM 64/128 - Two unassemblers.
SPEEDY UNASSEMBLER - An unassembler specific to Speedy Assembler users.
BANKS AND MEMORY - An aid to redifing screen and graphic memory.
GRAPHICS FACTORY - A novel way of getting in graphic design.
POT POURRI - A selection of useful routines for all users.

VOL 3 No.10 AUG '90
LIMBO 2 - Limbo No.2
SCREEN DESIGNER 128 - Easy screen designing.
DATABASE78 -Features galore for everyone.
LETTER MAKER - Pleasing text screens.
FUNCTIONS - Make full use of those function keys.
GAMES LIST CREATOR - Keep tabs on your games disks.
DUAL DISKCOPY - At last an intelligent disk copy program.
SEQUENCER64 - Musicians have a field day.
SECURITY - Put all those broken joysticks to good use.
SUPERBOOT! - Auto load your programs.

VOL 3 No.11 SEP '90
BANKING 128 - Keep your money in cheque.
DISK DRIVER V4 - A simple disk utility.
AUTOBOOT 128 - C128 users get easy access to CDU programs.
READING BETWEEN THE LINES - Build your own adventure parser.
I.D.O.S. -Comprehensive drive utility.
PRICE CALCULATOR - Keep tabs on inflation.
B.O.S.S. - Ext. Basic.
SCRN DES/COMP - Screen layouts for all.
LANDSCAPE ROUTINE - Beginners guide to scrolling backdrops.
SAMPLE KIT 64 - More sampling for all you musicians.

Back issues of CDU are available at £3.25 per issue, which includes postage and packing. All orders should be sent to:- Select Subscriptions Ltd, 5, River Park Estate, Berkhamsted, Herts, HP4 1HL. Please allow 28 days for delivery.

CONTENTS

Volume 3 Number 12 October 1990

IN THE MAGAZINE

ON THE DISK

Publisher: Hasnain Walji
Group Editor: Paul Eves
Technical Assistant: Jason Finch
Publishing Consultant: Paul Crowder
Consultant Editor: Stuart Cooke
Advertisement Manager: Cass Gilbert
Designer: Mark Newton
Distribution: S.M. Distribution
6, Leigham Court Road,
Streatham, London SW16 2PG.
Printed By: Gibbons Barford Print

Subscription Rates

UK	£33.00
Europe	£39.00
Middle East	£39.30
Far East	£41.60
Rest of World	£39.70 or $69.00

Airmail rates on request
Contact: Select Subscriptions. Tel: (0442) 876661

Commodore Disk User is a monthly magazine published on the 3rd Friday of every month. Alphavite Publications Limited, 20, Potters Lane, Kiln Farm, Milton Keynes MK11 3HF. Telephone: (0908) 569819 FAX (0908) 260229 For advertising ring (0908) 569819

EDITORS COMMENT

Isn't life full of unexpected and pleasing surprises. Only 2 months ago, it appeared that not only was my life ruined and in pieces, but that of all my loyal readers. The one and only C64 magazine devoted to the serious user was in danger of becoming no more. As it happens, things not only improved but also took a turn for the better. As you are now aware, we are living in the world of DTP. To the man in the street this might not mean much, but to us in the business of journalism, it means a whole lot. Not only can Editors now have more control over their publications, but they can keep a keener eye on how it will be finally presented. If your phone calls and letters are anything to go by, we have succeeded where all others have failed. From the evidence to hand, it would appear that every last one of you are happy with the new layout and design of your favourite magazine. In order to make sure we have everything spot on, you will notice there is a survey in this issue which I would like EVERYONE to fill in and return. By taking part, not only could you win yourself lots of lovely disks, but you will get the kind of magazine that you want. All those that have recently, since May, sent in submission to the magazine will be pleased to know that everything has now been evaluated and replies are winging their way to all those concerned.

This month's disk is a bit of a mixed bag. I have decided not to have a theme for a change, I hope you like it.

DISK INSTRUCTIONS

Although we do everything possible to ensure that CDU is compatible with all C64 and C128 computers, one point we must make clear is this. The use of 'Fast Loaders', 'Cartridges' or alternative operating systems such as 'Dolphin DOS', may not guarantee that your disk will function properly. If you experience problems and you have one of the above, then we suggest you disable them and use the computer under normal, standard conditions.

Getting the programs up and running should not present you with any difficulties, simply put your disk in the drive and enter the command

LOAD"MENU",8,1

Once the disk menu has loaded you will be able to start any of the programs simply be selecting the desired one from the list. It is possible for some programs to alter the computers memory so that you will not be able to LOAD programs from the menu correctly until you reset the machine. We therefore suggest that you turn your computer off and then on again, before loading each program.

HOW TO COPY CDU FILES

You are welcome to make as many of your own copies of CDU programs as you want, as long as you do not pass them on to other people, or worse, sell them for profit. For people who want to make legitimate copies, we have provided a very simple machine code file copier. To use it, simply select the item FILE COPIER from the main menu. Instructions are presented on screen.

DISK FAILURE

If for any reason the disk with your copy of CDU will not work on your system then please carefully re-read the operating instructions in the magazine. If you still experience problems then:

1. If you are a subscriber, return it to:
   Select Subscriptions Ltd
   5, River Park Estate
   Berkhamsted
   Herts
   HP4 1HL
   Telephone; 0442-876661

2. If you bought it from a newsagents, then return it to:
   Interceptor Group
   Mercury House
   Calleva Park
   Aldermaston
   Berkshire
   RG7 4GW
   Telephone; 0734-81742

Within eight weeks of publication date disks are replaced free.

After eight weeks a replacement disk can be supplied from Interceptor Group for a service charge of $1.00. Return the faulty disk with a cheque or postal order made out to Interceptor Group and clearly state the issue of CDU that you require. No documentation will be supplied.

Please use appropriate packaging, cardboard stiffener at least, when returning disk. Do not send back your magazine, only the disk please.

NOTE. Do not send your disks back to the above address if its a program that does not appear to work. Only if the DISK is faulty. Program faults should be sent to: BUG FINDERS, CDU, Alphavite Publications Ltd, Unit 20, Potters Lane, Kiln Farm, Milton Keynes, MK11 3HF. Thank you.

ROLL 'EM

Factory animation demo to show just what you can achieve with this Basic utility

MARCO WESTERWEEL

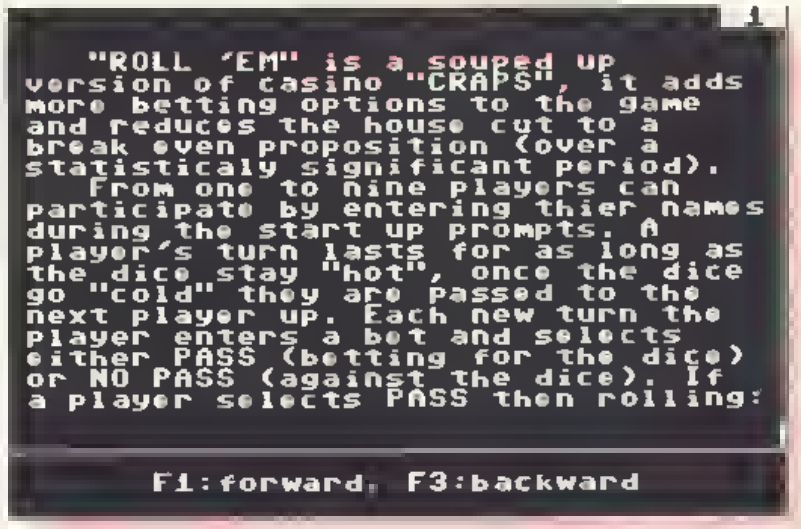
In JULY we published a great little utility for Basic programmers that wanted to have the world of graphics opened up to them. At the time, we said that we would publish a demonstratton of what could be achieved by using 'Graphics Factory'

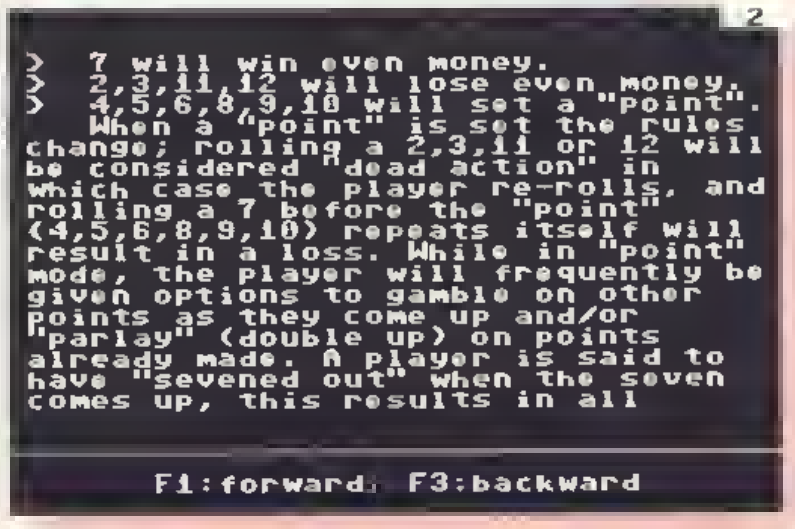
THE CONCEPT

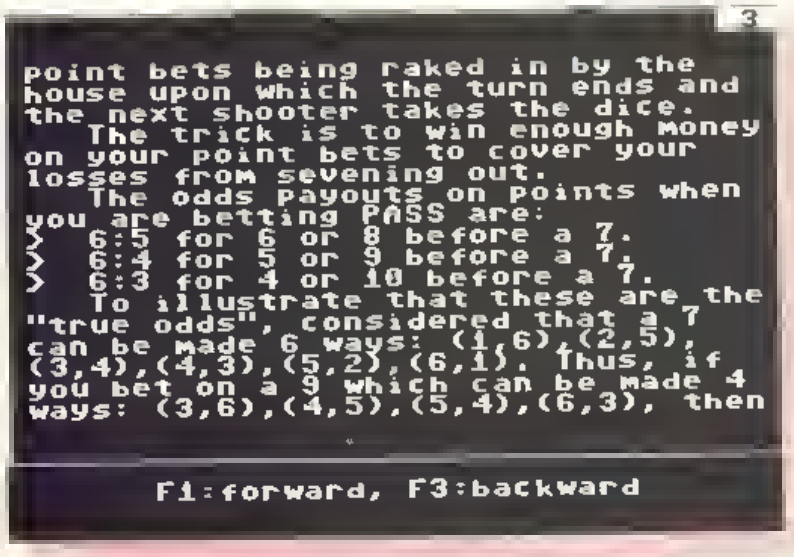
What is being demonstrated here is how to achieve animation in your Basic programs at respectable speeds. In this case the demo. program uses screens created with the 'Graphics Factory' screen editor, which as already mentioned, was published earlier. The animation technique described herein applies to any screen whether made with the 'Graphics Factory' utility or not

The trick to quality animation in Basic is to reduce the time a program takes to search the computer's memory for the graphics to be printed in various situations. This can be done by crunching string variables into arrays which store and index them as memory cells. Such cells can then be accessed without a slow sequential memory search, because the index numbers serve to instantly locate the appropriate graphic data cells for the program.

To give an example of how efficiently graphics arrays can work, I would like to refer back to the MAY 1989 issue of 'Your Commodore' which contained a 'Turbo Touch'







typing tutorial, which I wrote entirely in Basic. A variation of the above described method was used, and that program had a maximum process and print capacity of some 300 crunched strings per minute.

In this article, the emphasis is on graphics arrays in Basic programs that use the RND function for generating random numbers. The program in question is a dice game similar to CRAPS as played in casinos world wide. A few twists are added to eliminate the house advantage, and it makes the game more exciting by allowing multi-way betting action (watch out though, pyramiding your bets all over the board makes your bankroll fluctuate up and down as it it was riding a roller coaster). The rules can be accessed after RUNning the program, but a few words on strategy are included here. DON'T PUSH YOUR LUCK!!! While you can not influence the odds, you certainly can influence the amount of risk you take.

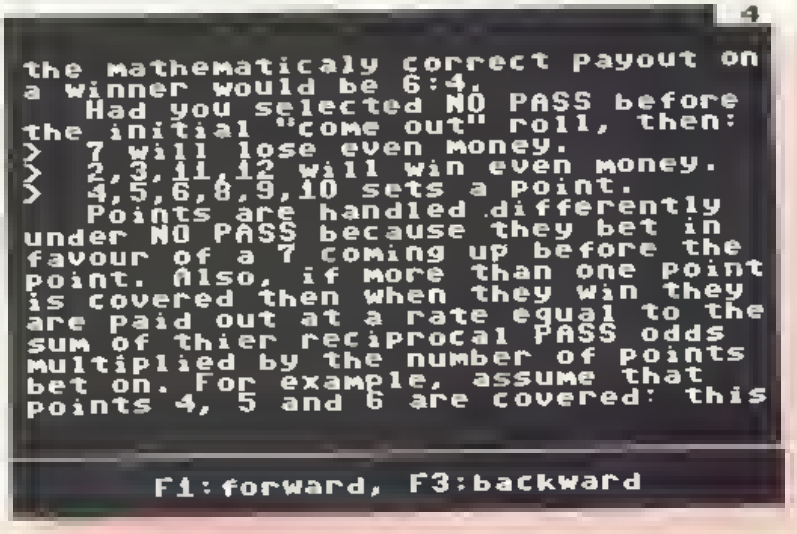
To follow the ensuing explanation on crunching strings into array, you must LOAD the program and LIST the lines described to the screen. The program has been liberaly documented with REM statements, the subroutines have been spaced apart with blank lines, and no program lines exceed 40 characters in length. Therefore the sections LISted to screen are structured in an easy to read format.

THE TECHNIQUE

Each string that is crunched into an array is a small program on its own. As long as you stay under the 255 character allowable maximum for a single string, you can assign that string's exact printing location, its colours, its composition and its dimensions all in one shot

Okay, so let's look at how this is done in the case of "Roll'Em". Type LOAD"**ROLL'EM**",8 and once loaded type LIST 1295-1365, which displays the initialisation routine. Line 1320 sets the maximum number of cells to be used in the arrays and assigns a random starting point to the random number generator. Lines 1325 to 1345 set up the screen background and border colours, and clears the sound registers. The numeric arrays in lines 1350 to 1365 contain the odds pay outs for dice totals equal to the numbers in parentheses.

The graphics crunching occurs between lines 1375 to 1620 in four different sections, but only the first two need be looked at for the purpose of this article. Section one (lines 1375 to 1430), replicates the squares in the game field layout There are 11 of them (2,3,4,5,6,7,8,9,10,11 and 12) arranged in two hemi-spheres. Line 1380 sets S1$ to represent a formatting variable which starts at the top left corner of the screen and extends 25 spaces down R1$ in line 1385 is also a formatting variable, it is made up of 36 spaces extending from left to right. The data in lines 1425 and 1430 represent various lengths of S1$ and R1$ which are read in line 1395 and assigned to S2$ and R2$ in lines 1400 and 1405. Lines 1410 to 1420 then create the graphics, add them to S2$ and R2$, and store them in array DT$(2 to 12). By storing the field squares in an array this way, the dice totals generated while RUN-ing the program can be used as index numbers for printing their corresponding squares in a distinguishing colour to indicate



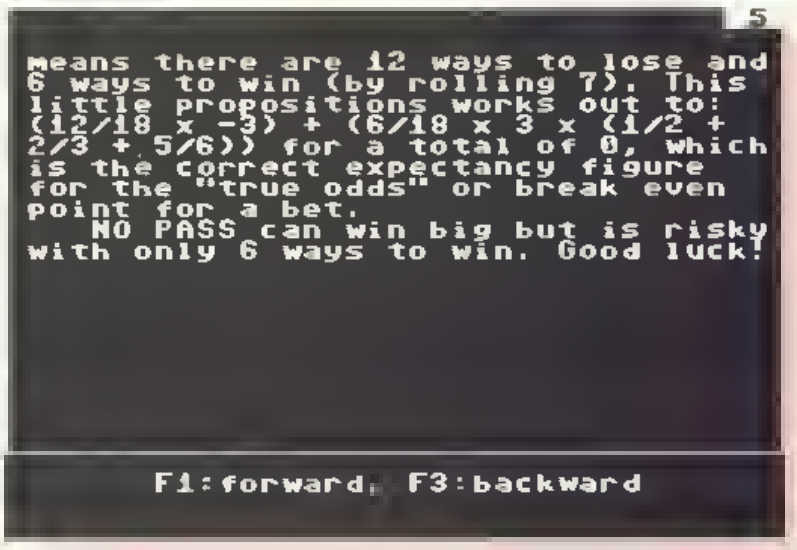
what happened, and reset them for a new roll

ONWARDS and ONWARDS

Next, we look at section 1400 to 1500 which is where the dice faces are compiled. Notice how lines 1445 to 1450 assign dots and blanks to string segments three spaces long, and how the last segment is made up of a formatting variable which prints one space down followed by three spaces to the left. The faces consist of three segments with the formatting segment inserted between each one to ensure that they print one above the other. Lines 1455 to 1480 connect the segments, then add various combinations of S1$ and R1$ (used in the previous section) in lines 1485 to 1495, and stores the graphics in arrays D1$(1 to 6) and D2$(1 to 6). These arrays are then used with the RVS ON/OFF/ON sequence in array V$(1 to 3) from line 1500 to animate the dice rolling subroutine.

ARRAY MANIPULATION

Having crunched the desired strings, it is now time to examine program section 880 to 945 to see how the arrays are manipulated to simulate a dice game concept. Lines 895 and 900 produce the random totals for each die. The two nested FOR-NEXT loops in lines 905 to 920 then print the array cells in a special sequence accompanied with rapid click-sounds from line 915 to create an overall "shake, rattle and roll" effect. The outside FOR:NFXT loop switches the RVS ON/OFF/ON pattern, while the inside loop rotates two patterns of dice faces in opposite directions. Since the last printing sequence leaves all dice faces in RVS ON mode, the closing loop between line 920 and 935 has the task of reconverting all but the two dice faces which correspond to the end total, back to RVS OFF. Lines 925 and 930 check to see if the array cells being reconverted are not the random dice totals from lines 895 and 900. The total is then assigned in line 940, and the appropriate field square is printed in reverse black in



line 945. Here the dice rolling subroutine is exited, upon which the game continues from wherever GOSUB 895 was last encountered.

IN CONCLUSION

The above routine performs quite a few functions at a very quick pace Basic can handle that because all of the formatting and composition work was pre-defined, and the program had fast access to the graphics in array form You will be able to achieve the same things in your own Basic programs by following the examples given here String crunching is easily applicable to almost any program based on board or table games. It can also be applied nicely to utilities such as operating menus in the form of multiple windows, shadow boxes or simplified flowcharts By using a screen designer such as my "Graphics Factory" program, you will be able to map out a program concept in which much of your animation requirements in the form of string crunching becomes self evident. Many other aspects of using this method are clearly documented in the program listing for "ROLL'EM", but the material covered in this article alone should get you well on your way to designing powerful programs using only Basic

ON THE DISK

COLOUR MATCHER 128

JASON FINCH

Sort out your colour schemes without sending your eyes on a psycadelic trip

Approximately two years ago a very similar program to Colour Matcher 128 was published for the Commodore 64. Because the utility, Colour Match, is useful when programming something that is to use colour, I find it surprising that it has had to wait until now for one to surface for the Commodore 128. And of course, who better to present it!!

For obvious reasons (the colours of the 128 screen in 40 column mode being the same as those on the 64) this program operates only in eighty column mode. There is no provision for black and white because this facility is not provided on most monitors in eighty column mode. My program also has a couple of extra options that the original, written by *DAVID BUTCHER*, did not

Just in case you did not see the original I shall explain what it does and how it will possibly help you. There are sixteen different colours which are not the same as those seen on the Commodore 64 or the 128 in 40 column mode. Thus there are 256 (16x16) combinations of character and background colours. The program will display the majority of these, in fact a total of 240, and then request you to press one of the numbered keys, whether it be from the top row or the numeric keypad. You press a number corresponding to your views of the combination and how it looks, the number being between one and four inclusive. A one indicates that you think the display is poor, for example dark cyan on dark green does not produce a good combination of colours, to me that is, because the text is not immediately easy to read. A two indicates that you think the outcome is fair or average. A three means good and if you think that the text really stands out well then press four for excellent.

The reason that the program only displays 240 combinations (besides your eyes going strange after only fifty!) Is because there are 16 combinations that produce no visible result, for example black on black, green on green and so on. Once you have entered your choice for all 240, a large grid will be displayed showing your results in tabulated form. This table is quite self-explanatory, the background colours being the numbers on the vertical axis, and the character colours those on the horizontal axis. For example, if you wanted to know how you reacted to light red text (colour 11) on a dark cyan background (colour 12) then you would look down the grid to the row of numbers with a twelve on the left and then read along until you came to the number in that row, but also in column number eleven. It really is very simple and all the colours and their numbers can be found on Page 6-5 of the C128 System Guide.

Once this table is displayed you have a number of options. The simplest being to press 'Q' to quit the program and return to BASIC. You could also press 'R' if you wished to restart the program and do all 240 over again (what a lovely way to spend a Saturday night that would be!!). There is also the option to dump the grid to a printer but this will be discussed a little later. If you press a number between one and four then the grid will be redrawn but all the combinations to which you assigned that number will flash. You could therefore easily see which combinations you thought were excellent, for example. To highlight another set of values simply press a different key and if you want to stop them flashing altogether then press zero. Finally, printing the grid. All you have to do is press 'P', align your paper and after pressing RETURN, away you go. Any numbers that were flashing at the time will be printed in reverse field on the printout. Therefore if, at some future date, you are writing a program and wish to see which combinations are good, then simply get the threes flashing and select print What could be easier than that??

I don't really think that any more explanation of the program is required. Simply switch the 128 on in 80 column mode or type GRAPHICS or use any of the other available methods to switch to 80 column mode. Of course you must have a monitor that supports this mode - unfortunately a standard television does not. To load it type DLOAD"COLOUR MATCH 128" and press RETURN. When the prompt reappears enter RUN. The first combination, white on black, will appear immediately. I hope you find this small and simple utility useful when writing your own programs on the 128.

STOP PRESS:-Before running the program, list it and add the start, 10 FAST

SPREAD-ED

A high quality spreadsheet program that compliments GRAPH-ED
on page 38 in this issue

Writing about **SPREAD-ED** is difficult as it is often hard to appraise months of your own hard work. I don't intend to make this a tutorial on how to use spreadsheets as there have been volumes published on this complex matter. In a single sentence, a spreadsheet is a matrix of cells which can hold labels, numbers and mathematical formulae which can reference other cells.

INTRODUCTION

SPREAD-ED is a simple but friendly spreadsheet program. It uses a pseudo-windowing technique to provide and request information. As well as being asthetically pleasing, it avoids the hideous screen clutter which most 8-bit spreadsheets exhibit. **SPREAD-ED** was primarily designed for managing home accounts, so the number of cells has been limited to 99 rows by 50 columns. It is unlikely that you will ever need so many cells, or that the 64's memory will stretch to these limits!

As always, the best way to learn is to experiment with the program. I recommend that you set up a simple spreadsheet for your household finances as this will force you to grasp the fundamental features of **SPREAD-ED**. You should not be able to crash **SPREAD-ED** and the program will normally ask for confirmation before destroying data, but I still recommend that you save your data regularly in case of unexpected results.

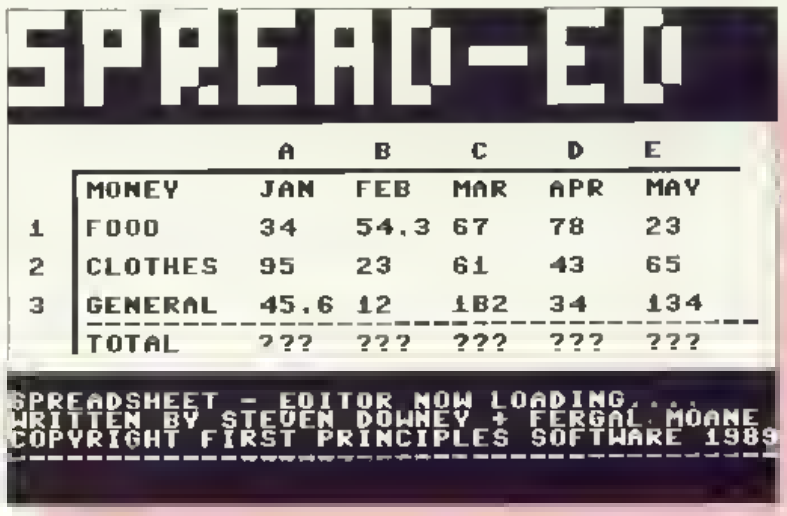
Perhaps the most efficient way of presenting the instructions is in a command summary. The most important functions come first in the list. Read what you need and get on with it! (some online help is available by pressing 'H' on the main menu.

COMMANDS IN SPREAD-ED

SPREAD-ED's first request is to load another file. As you have not created any spreadsheets, press 'N' for no. When you have some data saved, this is where you should load it back. This is because **SPREAD-ED** has to know how big the spreadsheet is and how much memory to allocate for it. Answering NO brings a prompt for the numbers of rows and columns. The limits for each are given besides the prompt. Initially, I would use a 15 by 15 spreadsheet. This is big enough to easily cover the twelve months of the year.

The main screen shows the matrix of cells. At the moment they are all empty, so dots appear in all spaces. The rows are numbered and the columns are allocated letters. This can give the co-ordinate of any individual cell by specifying letter and number EG; B4 or C2.

All the following commands are recognised from this main screen. The status bar at the top of the screen will show which command has been used.



CURSOR KEYS

The normal cursor keys will move around the **SPREAD-ED** cursor. This is two arrows which point to the

current cell. Most operations are dependent on the position of this cursor. Moving the cursor off the screen will scroll the spreadsheet in four directions allowing you to view cells which are not currently on the screen. Pressing 'G' will allow you to type in the row and column that you wish to go to. This is useful for hopping around a large sheet. This should only be used for cells that are not displayed on the initial screen (the top left). Pressing HOME will access the top left of the sheet.

NUMBERS

Typing any figure will allow you to enter a constant number in a cell. These numbers may be positive or negative with decimal points. Once you have finished entering the number at the current cursor position by pressing RETURN, **SPREAD-ED** will attempt to convert the number to standard form to save space. This is the workhorse of spreadsheets as they would be useless without numbers to work from. If you overwrite a number by accident, pressing 'O' will act as an undo function and return the last number.

RETURN KEY

As well as its normal function of terminating input, the return key will signal **SPREAD-ED** that you wish to enter a formulae or function in the current cell. This is what makes a spreadsheet so powerful. The calculations that can be entered are fairly simple, (add,subtract,divide,multiply,exponential etc), but it allows you to make What If? decisions. By altering one number in the spreadsheet, the effect can be carried forward through the rest of your budget. For example, imagine that column B contained all your expenditure on fuels for the year. By entering the following formulae B2+B3+B4*1.15 in say cell C3, after updating, Cell C3 will tell you the total expenditure on fuel and add VAT automatically for you. The great versatility in this is that cells with formulae can be linked. Elsewhere in the sheet, you can take the calue in C1 and subtract if from your total income next year. The cross-referencing of cells is a tricky programming technique, but provides sheets of great complexity that would otherwise be impossible by hand. Pressing RETURN will enter the formulae and Back-Arrow (top left) will exit if the option is chosen by accident

UPDATE SPREADSHEET

Updating is crucial to the working of **SPREAD-ED**. A single update is performed automatically after most functions which alter the sheet. Manual update is available by pressing F7 or 'U'. Cross referencing of cells can often catch the update process out. The update works from top left to bottom right (therefore the larger the sheet, the longer the update time). Working in this order can often mean that cells are referenced that have already been passed by. Updating the sheet a few times manually cures the problem. This is not a programming oversight - it happens on the best spreadsheets! Just remember to update a few times after changing anything drastic.

MANIPULATING FUNCTIONS

As functions/formulae contained in a cell can be long, it would be impractical to display the actual formulae in their cells. Pressing 'E' will examine the formulae contained in the cell underneath the cursor. Pressing 'L' will list all formulae in the entire sheet with their respective cell numbers to the screen or the printer. 'W' will wipe the formulae from the current cell. 'D' will duplicate the function from the current cell to the one you select with the cursor keys and return.

ORGANISING THE SPREADSHEET

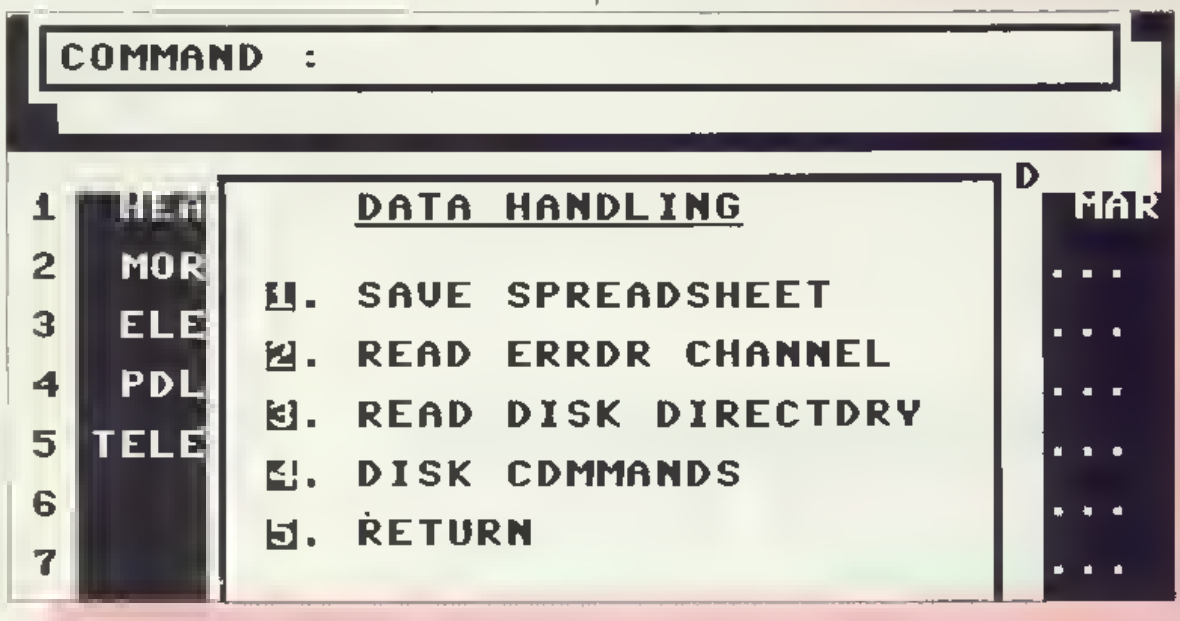
With all these figures, it is necessary to leave plenty of space between figures and place lables on the rows and columns. F2 will allow you to enter the title in the top left for the whole sheet. F3 and F4 will allow you to enter labels at the tope of each row and column respectively. These should be meaningful labels EG; January, Total, Mortgage. F5 and F6 will insert a blank row and column respectively. Pressing 'X' and 'Y' will insert a line along a row and a line along a column respectively. These are essential for dividing up a spreadsheet.

STORING THE SPREADSHEET

Pressing F1 will bring up the Data Handling menu. This allows you to perform disk commands, obtain a directory of the disk, read any errors and save the spreadsheet. The spreadsheet is stored in a sequential file, with all formulae, numbers and labels preserved. This file can be reloaded from the initial **SPREAD-ED** prompt. The file is also compatible with GRAPH-ED on page 38 in this issue. This program allows you to represent your numerical data in the form of graphs or charts. Pressing 'P' will produce a hardcopy of the spreadsheet on a Device 4 Commodore Graphics Printer. Unfortunately, you really need a sideways printing program to print large sheets successfully, but the present method often produces acceptable results.

FINAL POINTS

I do not claim that **SPREAD-ED** is the world's fastest or best equipped spread sheet. However, I feel that complex spread sheets are for complex accounts and as a home user I feel more at home with friendly **SPREAD-ED** than pouring over massive manuals that comes with lotus 1-2-3. Besides, at school my peers are learning about how spreadsheets work for GCSE and A-level. I have the satisfaction of knowing that I have written my own.



COMMAND SUMMARY

Cursor Keys	Move around Spreadsheet
Return	Enters formula in the current cell
Numbers	Place numerical values in the current cell
Home	Return to top left of Spreadsheet
Clr	Clear all values in Spreadsheet
F1	Data handling menu
F2	Corner label
F3	Row label
F4	Column label
F5	Blank row
F6	Blank column
F7	Update Spreadsheet
F8	Save Spreadsheet
P	Print Spreadsheet
L	List formulae to Screen or Printer
E	Examine formulae in current cell
D	Duplicate function from current cell
W	Wipe function in current cell
O	Return last number
X	Line along row
Y	Line along column
G	Go to cell
A	About SPREAD-ED and me
H	Produce online help screen
RUNSTOP/RESTORE	Reset SPREAD-ED

RASTER EDITOR

ROBERT TROUGHTON

EVERYONE CAN NOW PRODUCE THOSE COLOURFUL BARS WE SO OFTEN SEE ON EVERYONE ELSES PROGRAMS BUT OUR OWN.

With this utility you can quickly and easily make raster-bars for demo's, intro's, games and anything else you can dream up. In a previous issue of CDU, ***Andy Partridge*** told you how to use raster-bars, this utility enables you to design far more complicated routines with ease.

To give you an idea of the kind of effects that can be created, there are some examples provided on the disk. Just load the programs using the standard load routine. The names of the examples are Example Rasters 1 thru 4.

Just to show all you budding Raster Programmers how to actually use Rasters in your own programs, I have provided a short Machine Code routine on the disk that displays 128 raster lines at a time. It's a very simple routine and you can disassemble it with any of the monitors that are available. The 'set-up' routine is at $4E00 and the IRQ at $4F00. The raster colours are at $5000, which is exactly where they will load if you enter LOAD'filename',8,1. To see it in operation type LOAD"Raster-Demo $4E00",8,1 then SYS19968.

I hope that you enjoy using this utility and that it gives you many hours of fun.

USING THE EDITOR

KEY CONTROL	EFFECT
CRSR (u/d)	Move Cursor
SPACE	Clear Raster line
0-9, A-F	Place Raster colour
INST	Insert line
DEL	Delete line
CLR	Clear whole raster memory and home cursor
HOME	Home cursor
RETURN	Carriage return
+	Move up through raster memory
-	Move down through raster memory
BACK ARROW	Set marks
CBM C	Clear area set by marks
CBM F	Fill area set by marks (select colour)
CBM I	Invert area set by marks
RUNSTOP	Clear marks
CBM D	Directory
CBM S	Save rasters (specify filename)
CBM L	Load rasters (specify filename)

READERS SURVEY

Complete our readers survey and you could win 25 blank disks!

In order to gauge exactly what you, the readers, would like to see in future issues of CDU. Would you be kind enough to answer our few brief questions about the magazine.

As an added incentive we will be giving away five packs of twenty five blank disks, with the distinctive CDU logo, to the first 5 completed coupons pulled out of the hat on the closing date

Once you have completed the coupon just pop it in the post and send it to:

CDU Readers Survey
Alphavite Publications Ltd
20, Potters Lane
Kiln Farm
Milton Keynes
MK11 3HF

Completed coupons should arrive no later than December 31st 1990.

1 How long have you been a CDU reader?

Less than 3 months ❑
3-6 months ❑
7-12 months ❑
13-24 months ❑
25-34 months ❑

2 How often do you buy CDU?

Occasional issues ❑
Most issues ❑
Every issue ☑

3 Does anyone else read your copy of CDU Y ☑ N ❑

4 If yes, how many?
1 other ☑ 2 ❑ 3 ❑ 4+ ❑

5. How much of CDU do you read?

Read only some articles ❑
Read most articles ❑
Read all articles ☑

6 Which of the following would you most like to see featured with the magazine? (please tick only one box).

Cover mounted gifts ❑
Additional supplements ❑
Competitions ❑
Money saving offers ☑

7 With respect to the articles/programs in CDU, how do you rate the following?

	POOR ❑	AVERAGE ❑	GOOD ❑	EXCELLENT ❑
Games programs	❑	❑	☑	❑
Utility programs	❑	☑	❑	❑
Programming features	❑	❑	❑	❑
Utility reviews	❑	❑	❑	❑
Hardware reviews	❑	❑	❑	❑
Competitions	❑	❑	❑	❑
Other features	❑	❑	❑	❑

8. Which other computer magazines do you read and how often?

	NEVER ❑	OCCASIONALLY ❑	REGULARLY ❑
YC	❑	❑	❑
Commodore User	❑	❑	☑
Zzap!	☑	❑	❑
Amiga User Int.	❑	❑	☑
Your Amiga	❑	❑	☑
Amiga Computing	❑	❑	☑
ST/Amiga Format	❑	❑	☑
Compute	❑	❑	☑
Compute's Gazzette	❑	❑	☑
Other (please specify)	❑	❑	❑

9. If read, how do they compare with CDU?

	NOT AS GOOD AS CDU A	AS GOOD AS CDU B	BETTER THAN CDU C
YCCommodore User.			
Zzap!			
Amiga User Int.			
Your Amiga			
Amiga Computing			
ST/Amiga Format			
Compute			
Compute's Gazzette			
Other (please specify)			

10. Are you aware of the scheduled publication date?

YES ☑ NO ❑

11. If the answer to question 10 is YES, do you attempt to purchase the magazine that day?

YES ☑ NO ❑

12. How do you normally obtain your copy?

Chance purchase ☑
Newsagent shop collection ❑
Newsagent home delivery ❑
Subscription ❑
Passed on copy ❑

13. If you are a subscriber, on which date did you receive this issue?

/ / /

SURVEY

14. If you do not obtain your copy by subscription, is it due to one of the following?

Subscription too expensive ☑
Not every issue required ❑
Not aware of subscription service ❑

15. Are you aware that to subscribe to this magazine in the U.K. is the same cost as purchasing it in a shop?

YES ☑ NO ❑

16. Would you like further details on taking a subsription?

YES ❑ NO ❑

17. Which disk drive do you own?

1541 ☑
1570 ❑
1571 ❑
Other Commodore ❑
Excelerator + ❑
Enhancer 2000 ❑
Other ❑

18. Which model of computer(s) do you own?

C64 ☑
C64C ❑
C128 ❑
C128D ❑
Plus/4 ❑
C16 ❑
Amiga 500 ❑
Amiga 1000 ❑
Amiga 2000 ❑
Amiga 2000B ❑
Amiga 3000 ❑

19. Do you have a speed-DOS device fitted?
(e.g. Dolphin DOS)

Yes ❑
No ☑

20. What will the next purchase for your Commodore Computer system be?

....
....
....
....
. ...

21. Where do you usually buy your software?

Specialist outlet ❑
Mail order ❑
Chainstore ❑
Other ❑

22. Please enter a few personal details below to help us find a little more about our readers. These details will be in the strictest of confidence. No information will be given to any other parties whatsoever. Don't forget to add your name and address if you would like to be entered into the draw for the free disks.

Sex M ☑ F ❑

Age
Under 15 ❑
15-18 ☑
19-21 ❑
22-24 ❑
25-34 ❑
35-44 ❑
45-54 ❑
55-64 ❑
65+ ❑

23. Are You...?

In full-time employment ❑
In part-time employment ❑
Not employed at present ☑
Student - full-time ❑
Student - part-time ❑
Retired ❑

24. If in full-time employment, please state your occupation.

....

25. Do you think that CDU has got the format right or would you like to see more or less of the following items?

	LESS	O.K	MORE
General features	❑	❑	❑
Games Reviews	❑	❑	❑
Programming features	❑	❑	❑
Business features	❑	❑	❑
Utility/hardware reviews	❑	❑	❑
Book reviews	❑	❑	❑
Games programs	❑	❑	❑
Utility programs	❑	❑	❑
Educational programs	❑	❑	❑
Business programs	❑	❑	❑
Music	☑	❑	❑
Sound programs	❑	❑	❑
Competitions	❑	❑	❑
News	❑	❑	❑

To enter our FREE draw, fill in your name and address details and remember, all entries must be returned by 31st December 1990.

Name.....

Address.....
.....
.....

(only if you want to enter the draw)

Should you want to enclose any comments about CDU, or offer any suggestions, then we would be only too pleased to receive them with this survery. (Please enclose on a sepearate sheet).

Finally, I would like to thank you for taking the time to participate in this survey. From the information you the readers provide, we can continue to make improvements in the quality of service that CDU provides for it's many thousands of readers.....Ed!!

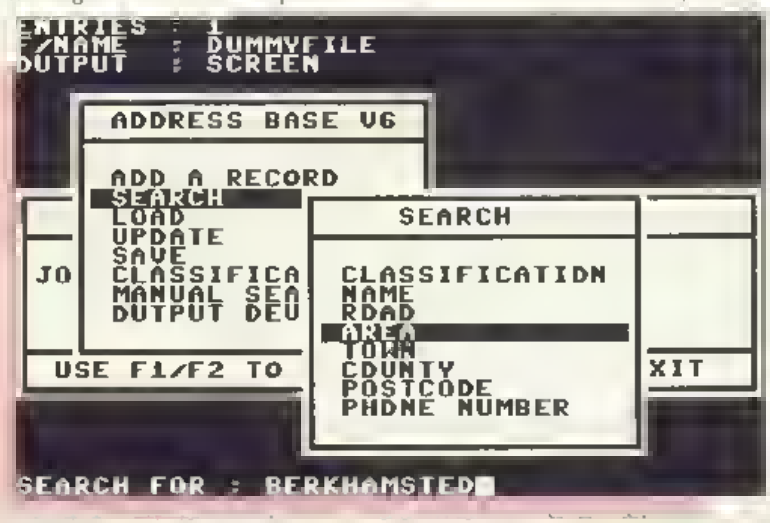
Address base

All address bases are not the same as using this program quickly becomes obvious

MARK SKINGLE

Address base is an address book with a difference. The program allows you to store *500* names, addresses and telephone numbers, along with a classification. You could use this facility to distinguish some addresses from others.

When the program has loaded, you will be greeted with an information window. Press *RETURN* and the screen will clear. At the top left of the screen is a general information panel.



ENTRIES tells you how many addresses are stored in memory at the present time. *F/NAME* is the current filename and *OUTPUT* tells you whether the data will go to the screen or printer.

USING THE PROGRAM

Address Base uses window menus which makes it very easy to use. To select an option use the *F1* (up) and *F3* (down) keys to move the select bar and press *F7* to select the desired function. Move to *LOAD* and press *F7*, at the top of the screen *F/NAME* will turn red and a cursor will appear in the place of the old filename. Type in *COMPUTERS* and press *RETURN* the data file will start to load. *COMPUTERS* is a file dedicated to containing addresses of computer companies only. You could have one for relatives, friends, etc. Each of these files can hold up to *500* addresses

Once the file has loaded select *CLASSIFICATIONS*. A window will appear to show the classifications in use. You can scroll through these using the function keys. Press *F7* to return to the main menu. To illustrate the use of the classifications, select *SEARCH*. Now select *CLASSIFICATION* as the search field. At the bottom of the screen a prompt asking for the *SEARCH DATA* will appear, type in *SOFTWARE* and press return. The companies dealing with software will now be listed. Press *'P'* to pause the listing. Now select *OUTPUT DEVICE* and then PRINTER. Begin to get a printout, using the search facility, of all the companies based in London then do the same for those that are dealing with software. Select *UPDATE* and *RECORD*, the output will automatically change to the screen, and type in the record number 1. The contents of this record will be dumped to the screen. Now alter Fred Blogg's address, (for those fields that you wish to keep the same just hit *RETURN*). Next, use the search facility to see that the record has been updated. Select *UPDATE* once again, followed this time by *FILE*. The updated file will be copied back onto the disk under the same name. To store the file under another name the SAVE option should be chosen from the main menu.

To view all the records select the *MANUAL SEARCH* option from the main menu. Use *'P'* to pause the listing (pressing *'P'* continuously will step through the listing one by one). Use *'Q'* to quit the listing. You can use this function to get a complete printout of all the records if *PRINTER* is selected as the output device.

ADD A RECORD can be used to tag another record onto the end of the file, allowing the file to get progressively larger. A very useful facility is the Sort routine, accessible via the update menu. To sort the records into alphabetical order, by name, select *NAME*. To place the records into order by classification, with a sub-order by name select *CLASS*. It is possible to use the program like a computerised 'Thompson local directory', the difference being that this program is not restricted to local addresses.

LOADING

To load the program either select *ADDRESS BASE* from the main menu or load by the standard load routine of *LOAD"ADDRESS BASE",8*. If you use this option, you will be able to *LIST* the program out. On doing so you will notice that the program has been compiled using *"PETSPEED"* by *SYSTEMS SOFTWARE (OXFORD) LTD*. *PETSPEED* saves the array and variables along with the program, it is for this reason that the program takes up so much disk storage space.

FEATURE

BASIC EXPLORED

Finding your way around a Basic program whilst it is in memory is made easier to understand by this short article.

S.WICKHAM

The purpose of this article is to give the novice programmer, a better understanding of the structure of a BASIC program within the computers memory, without being too technical or high brow. I shall try to keep in mind that it is for the novice, therefore I shall try to keep things simple. There may be certain areas where I have assumed that the reader will know what I am talking about. Mainly in terminology of phrases/words. With perhaps mention of locations within the computer. If you come across something that you are not sure about. There are plenty of referance books or articles you can check through for clarity. (The serious' users guides of 1987 and 1988 for example).

A SHORT INTRODUCTION TO BASIC:-

As you will no doubt know, BASIC stands for:- Beginners All-purpose Symbolic Instruction Code. It was implemented with the idea that the 'masses' could have access to a simple to learn and easy to use programming language. To this end the language is one hundred percent succesful. However, as we all know, it does have certain limitations. (Especially the version implemented on the C64). The idea of this article though, is not to run down the language, but to give you a better understanding to it's structure within the computers memory, with some hints and tips on how to get the most out of the limited amount of memory available for your programs.

LOOKING AT CODE IN MEMORY:-

When you are typing in your program, you may wish to view some section of it at some time. To do this you give the instruction LIST, immediately your program begins running past your eyes from start to finish. obviously, what you see on the screen and what is in the computers memory are two differant things altogether. Indeed, the lines flashing past your eyes have undergone many changes from it's journey from memory to screen. So just how do we get to view the program in memory. As we all know, on power up the default setting for the start of BASIC memory is 0801 HEX (2049 Decimal). So to view lets say the first 50 bytes of memory you would type something like the following:-

FORA=2049TO2099:PRINTPEEK(A);:NEXT. This would be done in direct mode. What you would now see is a series of decimal numbers in the range 0-255. These numbers are the decimal equivalents to the HEX numbers that are stored in memory. These HEX numbers are the machines interpretation to what you typed on the keyboard. To give you a better understanding of this look at figure 1. This is a simple 7 line basic program. Now look at figure 2. This is how that 7 line program is stored in the computers memory. I shall now explain how the program is made up line by line.

Each and every BASIC line has an overhead of 5 bytes, plus the line itself. These 5 bytes consist of a link address, line number and a trailing zero byte, which denotes the end of a line. The link address is used by the system when your are using the Screen Editor, that is to say, when you are inserting or deleting lines of a BASIC program. They are ignored when the program is RUNning. (More about the link addresses later). Back to our short example

```
10 POKE53280,1:POKE53281,2
20 PRINT"{CLR}
30 PRINT"HELLO EVERYONE"
40 FORA=1TO25
50 NEXT
60 PRINT"END OE DEMO"
70 GOTO70
```

FIGURE 1

So what does all this mean? The first thing that may spring to your mind is this. How does the computer know how to pick out the COMMAND words. The answer is simple. Any Command word, or KEYWORD word as they are called, has a HEX value in the range 7F to CB. When you press the return key the data is sent to the keyboard buffer for interpretation. A look up table is then used and your line is scanned for the keywords. The more observant of you will be asking yourself. What if I type the following: PRINT"PRINT". Wouldn't this be interpreted as 2 print statements. The answer is no, one of the jobs of the look up table is to check on the quotes mode. If a keyword lies between quotes then it MUST be WITHIN a print statement. Back to

(Remember all the values you see here are in HEXadecimal notation, not decimal. If you look in the Serious Users Guide, there is a conversion table if you wish to convert to decimal).

```
MEM  HEXADECIMAL NOTATION AS  ASCII OF ADDR  REPRESENTED
IN MEMORY  HEX'IMAL
0800: 00 17 08 0A 00 97 35 33 '. ....53
0808: 32 38 30 2C 31 3A 97 35 '280,1:.5
0810: 33 32 38 31 2C 32 00 20 '3281,2.. 
0818: 08 14 00 99 22 93 22 00 '....".".
0820: 36 08 1E 00 99 22 48 45 '6....."HE
0828: 4C 4C 4F 20 45 56 45 52 'LLO EVER
0830: 59 4E 4E 45 22 00 42 08 'YONE".B.
0838: 28 00 81 41 82 31 A4 32 '(..A.1-2
0840: 35 00 48 08 32 00 82 00 '5.H.2...
0848: 5B 08 3C 00 99 22 45 4E '[.(.."EN
0850: 44 20 4F 46 20 44 45 4D 'D OF DEM
0858: 4F 22 00 63 08 46 00 89 'O" ..F..
0860: 37 30 00 00 00 ..... '70.....
```

FIGURE 2

our little example

Memory address 0800 contains 00. For Basic to operate correctly, the memory address immediately before the start of BASIC memory must contain a zero byte. The next two bytes contain 17 08. These represent the link address in the usual low byte/high byte format. These two bytes tell the computer where the next basic line starts from. The next two bytes show 0A 00, this is the line number shown once again in the low/high byte format. The next byte is 97. Remembering what I said about keywords falling in the range of 7E and CB, signifies that this is a keyword. In fact its the hex notation for POKE. The following 5 bytes are 35 33 32 38 30. This represents the number 53280. The following byte is 2C, which is interpreted as a comma. Look at the rest of the example, and see if you can follow it through line by line. At first it may seem strange to you. With practice though, you will find that it becomes more and more clearer. (It would help you to have a table of CBM64 Tokens handy. You may be asking wondering, how does the computer know when it comes to the end of a BASIC program. Remember that link address I mentioned. Well if the link address contains a double zero byte then that marks the end of the program. As a matter of interest If you wanted to know the start and finish of a BASIC program you can always PEEK the following locations. $2D/2E and $2B/2C. These zero page locations are known as VARTAB and TXTTAB respectively. Going back to our example. If you look at FIGURE 2, you will see that location 0862 marks the end of our little demo program. If you count the number of bytes you will see that this short 7 line program takes up 97 bytes. It dosn't take much to realise that BASIC consumes memory very quickly.

So what can we do to get the most of the available memory. As you may know, all the keywords can be abbreviated. In most cases this means typing the first letter as normal, then typing the second letter only in shifted mode. Occasionaly, you need to type the first two letters, then the third as shifted. The one exception to this rule if the PRINT statement. This can be abbreviated by the ? key. For example ?"HELLO" is the same as PRINT"HELLO". Likewise PshiftedO is the same as POKE. (Again the user guides give a full list of abbreviated keywords). So why does this help us. The answer is simple. The less number of lines you have in your program the less number of bytes you take up, (Remember, each line has an overhead of 5 bytes plus the line itself). The reason for using shifted keywords is this. Although the screen editor on the 64 is perhaps the best there is. It does limit us to only 80 characters per program line. That is to say, two screen lines. Therefore, by using shifted keywords we can squeeze more instructions per program line. The second way of gaining more memory, is by putting more than one instruction on a line. The best way of demonstrating this is by example. Figure 3 is our 7 line demo retyped incorporating the two above tips. Figure 4 is the new way that it is stored in memory.

```
10  POKE53280,1:POKE53281,2:
    PRINT"HELLO EVERYONE":
    FORA=1TO25:NEXT:PRINT"
    END OF DEMO"
20  GOTO20
```

FIGURE 3

```
MEM  HEXADECIMAL NOTATION AS  ASCII OF ADDR  REPRESENTED
IN MEMORY  HEX'IMAL
0800: 00 43 08 0A 00 97 35 33 '.C....53
0808: 32 38 30 2C 31 3A 97 35 '280,1:.5
0810: 33 32 38 31 2C 32 3A 99 '3281,2:.
0818: 22 93 48 45 4C 4C 4E 20 '".HELLO
0820: 45 56 45 52 59 4E 4E 45 'EVERYONE
0828: 22 3A 81 41 B2 31 A4 32 '":.A.1-2
0830: 35 3A 82 3A 99 22 45 4E '5:.:."EN
0838: 44 20 4F 46 20 44 45 4D 'D OF DEM
0840: 4E 22 00 4B 08 14 00 89 'O".K....
0848: 32 30 00 00 00 00 00 00 '20.....
```

FIGURE 4

REMEMBER IN REALITY ALL THE KEYWORDS ARE ABBREVIATED, BUT FOR CLARITY ARE SHOWN AS NORMAL.

Now if you count up the number of

bytes you will see it comes to 73. So already we have made a saving of 24 bytes. And thats only on a very short program. Imagine the savings you could make on a large scale program. Another area we can save memory on is our variables. It is important to know what types there are, how they are made up, and most important of all, how they are stored in memory.

There are three types of variables allowed in BASIC. String, Integer and Real. If you include Function names, there are four. String variables have the dollar sign after the name. Integer, have the percent. And Real have nothing. The default value is Real. At this point, I will assume that you know how to make up a variable. Also the rules on what you can and can not include in the name. Our main aim is to see how they are stored in memory.

All variables are stored immediately after the BASIC program in memory. They are also stored in the order of creation. Strings however have two pointers. One is the address of the string, and the other is the strings length. Strings are also stored at the top of BASIC memory and work downwards. Variables are either SIMPLE or SUBSCRIPTED. Simple variables use an overhead of 7 bytes, made up as follows:-

The first two bytes hold the variable name. For Strings the next byte holds the length of the string. The next two it's pointer in low/high format. The remaining two are unused Integers use bytes three and four for it's sign and value, with the remainder unused. Real uses byte three as it's expression, with bytes four to seven as it's sign and mantissa. A function also uses seven bytes, the third and fourth point to the defination. The fifth and sixth point to it's variable and byte seven it's intitial value.

Subscripted variables, unlike the simple ones, only require the associated values for storage. For example, String subscripts only require three plus the length. Integers only need two.

Earlier on, I talked of ways of securing yourself as much memory as possible, by using abbreviated keywords. Putting as many instructions on one line as possible. I will now give you one or two hints on how to make your BASIC program RUN a little quicker. The obvious one is that the less line numbers there are, the quicker the program will be. Secondly, ensure that all important variables are declared early on in the program. This will save look-up time. Third, try not to have thousands of GOTO's or GOSUB's All that going and coming takes time. Finally, if you have lots of DATA, put it at the FRONT of your program. This will save time on each READ.

I hope that this article has helped you to understand just how BASIC is stored. I have made it as brief as possible so as not to over confuse the novice. The best way of learning is through practice. Therefore get yourself a machine code monitor, and start looking around. It will surprise you, just how much knowledge you gain by POKING and PEEKING around memory. Have fun...!!!

SUPERSORT 64 & SUPERSORT 128

Sort those strings out without bloodshed or tears

MIKE GREGORY

Supersort is a very useful utility which will sort alphanumeric strings within a sequential file It gets its name from originally being written to work with files from the popular word-processor 'SUPERSCRIPT'. Since Superscript files are normal Commodore sequential file format, Supersort will work with any other word-processor that uses this format, for example, ***Easyscript***.

Supersort allows you to use your compatible word-processor as a mini-database. Mailing lists and other address lists are good examples of the sort of information that often does not require a fully fledged database. The major advantages of using your word-processor are that you only need to learn one set of commands, and presumably you have already learned these, and also you have the full editing power of the word processor

HOW DOES IT WORK?

The utility has some advantages over other sort programmes. First of all, it is fast. Very fast. It uses a sort algorithm known as a 'Quicksort'. The method works with a divide-and-conquer principle. Think about sorting a pack of playing cards into numerical order and suits. You can do it by working directly on the whole pack, but it is very much quicker to separate the pack into the four suits first and then put each suit into numerical order! This is how a

Quicksort works. For those mathematically minded, it can be shown that the time required to sort N records is proportional to NlogN. This can be compared with, for example, a 'Bubble' sort which is proportional to N squared!

Supersort also puts numbers into numerical order. To understand this remark you need to appreciate that most sorts are done on the basis of the ***Ascii*** value of each byte or character. These values range from 65 to 90 for A to Z, from 48 to 57 for 0 to 9 and a split range of 32 to 47 and 58 to 64 for the normal punctuation marks and other symbols. As an example, sorting on ***Ascii*** value, the records,

1A
2A
10A
and 20A

would be sorted as,

10A
1A
20A
2A

in what is laughingly called ascending order! Try it with your favourite sort program if you don't believe me. ***Supersort*** will put the records into the proper ascending order.

Supersort allows you to select the position within a record where you want the sort to start. You are not tied down to the first character. You can select, for example, the seventh and ***Supersort*** will ignore the first six. There is an inbuilt safeguard here in that ***Supersort*** will over-ride a selected position which is greater than the shortest record. It will reset your chosen position to the last character in this shortest record! This ensures that every record always has a value.

HOW IS SUPERSORT USED?

The utility is very simple to use. Having prepared your address list or whatever, it is saved, as a sequential file remember, in a file called

SORTFILE

This name is FIXED. ***Supersort*** looks for this file to find the records to be sorted. Within the file, you pass information to ***Supersort*** to tell it the filename to which the sorted records are to be written and also which position is to be selected for sorting. After preparing and saving SORTFILE, all you have to do is load and run ***Supersort***. ***Supersort*** does everything else for you. The name of the file for the sorted records is given as,

***Userfilename
***Sortposition

and ***Supersort*** scans through SORTFILE for the three asterisks, remembers the supplied name, then grabs the sort position and goes on to the sort. The records should immediately follow, on the next line, the two asterisked control lines as shown above. It perhaps should be noted that, for our purposes, a record is all the text from the start of a line up to a carriage return/linefeed, with a maximum length of 254 characters. Each record must start on a new line immediately after the preceeding record.

All of this is much easier to follow in an example. Use your word processor to enter the following data. I have numbered the lines for reference (L01- etc). You should not enter these values, your lines should start at the character immediately after my line number. For example on line 3, at the 'F'.

```
L01-
L02- 1                15              29
L03- First Name       Second Name     Occupation
L04-
L05- ***second
L06- ***15
L07- Paul             Eves            Editor
L08- Stuart           Cooke           Group Editor
L09- Hilary           Curtis          Production Editor
L10- Alan             Batchelor       Cartoonist
L11- Manny            Cefai           Photography
L12- Gordon           Hamlett         Adventure Correspondent
L13- Paul             Kavanagh        Advertisement Manager
L14- Maria            Wade            Display Sales Executive
L15- Tony             Flanagan        Classified Sales Executive
L16- Mark             Newton          Designer
```

The setup shows how I have placed the information into fields just as in a database. In line two, I have shown the sort position numbers which correspond to the start of each field. The two control lines are set for sorting on the second field (surnames) and saving the sorted file as "Second". After entering the data as above, it should be saved as SORTFILE. Run ***Supersort*** and then examine the new file called SECOND. You will see that the names are now in surname order and that the control lines have been removed. All the other lines are still there. Any word processor settings for the printer and such which would normally be at the start of the file will also be retained. The only lines which are moved are those which are after the control lines. If you re-insert control lines

L05-***job
L06-***29

and re-save SORTFILE, you can run ***Supersort*** again and sort the database this time on field three.

I'm sure that as you use ***Supersort***, you will start to find more and more uses, you may even wonder how you managed without it! For example you can catalogue your disks and edit out all of those part-programmes which are only supporters to the main event.

FEATURE

FURTHER ADVENTURES IN C

JOHN SIMPSON

Our excursion into the world of 'C' takes a few more steps forward.

When we construct a program, we often meet with a series of logical decisions; if ... else. This is where the switch ... case statement comes into its own. Rather like a multiway decision maker. The switch causes control to be transfered to one of a multiple of statements dependant upon the value of an expression.

When a switch statement is acted upon, or executed, it's expression is evaluated and compared with each of a multiple of case constants. If one of the case constants is equal to the value of the expression, then control will pass to the statement following the case prefix.

To trap any possible errors, after the last case constant, and before the closing brace, we can place;

```
default:
        (action);
        break;
```

In other words, should the expression of switch not equal any of the case constants, then control would pass to default and execute any action held there.

In the switch ... case everything within the braces is going to be tested. Note the use of the colon after the case constant - this is compulsory. Note also the break statement. Let's say that the first case constant was equal to the switch expression, then the appropriate action would be executed and the next statement, break, would be encountered. This would cause control to pass to the end of the block, or closing brace. If you omit the break, then control will 'fall through' to the next case statement.

In some cases this would be necessary. For example, if you wanted to discover if the current key press on the keyboard was a vowel, or some other key you could use something like the following ;

Because we require the output from the keyboard, we must include the library function which deals with input/output, this is the first line.

#include <stdio.h>

When the compiler does its work, the first thing it will do will be to get the necessary file from the functions library and include it within the program. The next statement is main() and the setting up of the function with an opening brace [{]. We then assign an integer variable to k - int k. Next comes a simple print statement telling the user what to do, which is followed by a while ... loop.

while ((k=getchar())!='@')

The expression of the while loop is that k is set to equal the current keyboard response - getchar() - this is the library function which places the key struck into the keyboard buffer. If it does not equal the '@' key the program drops through to the switch. The symbols [!=] represent not equal to, (the same as Basics <>).

Now we come to switch(k). The value of the key response is an ASCII integer value (65 in the case of 'a'), which is placed into our integer variable k (for key). This will now be tested with the first case constant - 'a', and evaluate if k is equal to 'a', and so on through the five case constants. If any of them is true, control will pass to the printf statement, and then break from the block back to the while loop.

If the value of k does not equal any of the five case constants, control will pass to default and execute the print statement there, after which it

```
EG;  switch (expression)
     {
      case 1:
            (first action);
            break;
      case 2:
            (second action);
            break;
     }
```

```
#include <stdio.h>
main()
{
int k
        printf("type in a letter  <'@' to
stop>\n");
          while ((k=getchar())!='@')
          {
          switch (k)  {
          case 'a':
          case 'e':
          case 'i':
          case 'o':
          case 'u':
       printf(" - is a vowel\n");
       break;
          default:
       printf(" - is not a vowel\n");
                    }
          }
}
```

will 'fall through' to the closing brace, and return to the while loop.
Another situation where allowing control to 'fall through' a case constant may be where the program requires a user input of either an uppercase/lowercase letter within a menu, for example,

```
case 'A':
case 'a':
        menua();
        break;
```

```
/* lesson 8 ... the switch case tree . */
#include<stdio.h>
main()
{
   int key;
   printf("\n\n          MENU");
   printf("\n          LIST FILES");
   printf("\n          ADD A FILE");
   printf("\n          DELETE A FILE");
   printf("\n          VIEW FILE");
   printf("\n          EDIT FILE");
   printf("\n       QUIT  PROGRAM");
    printf("\n\n\n\n   <please select
option>");
            while     ((key=toupper
(getchar()))!='Q') {
        switch (key) {
          case 'L':
                printf("\n You selected
'L');
             break;
          case 'A':
             printf("\n You selected'A');
             break;
          case 'D':
                printf("\n You selected
D');
             break;
          case 'V':
             printf("\n You selected'V');
             break;
          case 'E':
                printf("\n You selected
'E');
             break;
          case '\n':
             break;
          default:
             printf("\n No such item in
the MENU");
             break;
          }
      }
}
```

Talking of menus is a neat example of a situation where the switch ... case is put to very good use. The programon the left - lesson 8 - will show a demonstration of this.

INTEGER USEAGE

Because we are going to use an integer variable entered from the keyboard, we will require the use of a standard identifier word for reading that input, this is getchar(), which is a function that required the use of the stdio.h header. So we must have the line of code: #include<stdio.h>: at the start of the program before any functions are defined.
After this we define the first function block as main() and the opening brace for the main() function. Next step is to assign the label key as an integer type variable - int key. The next section of code is some pretty obvious print statements for the MENU box. After the MENU box we set up our while loop in the form:
while((key=toupper(getchar()))!='@')
During compile time the compiler will evaluate the innermost of any nested loops, in the above statement;
1st - (getchar())
this reads the current keypress and assigns it to getchar.
2nd - (key=toupper(getchar()))
the expression toupper converts any lower case letters into upper case letters, and assigns the value gained from getchar into our variable, key.
3rd - (!='Q')
the symbol != represents not equal to and the expression states that if the value of key does not equal the letter 'Q' then continue with the loop, otherwise exit to the brace marking the end of the while block - which will then exit to the end of main() function block where the program will terminate. However, if a key other than 'Q' has been struck the control will pass on to the next stage which is;
switch (key)
The switch evaluated the integer expression in parentheses (in this program, key), and then compares this value to each case statement which follow. The case must be labelled with an integer or character constant, or constant expression.
If a case matches the value of the expression (key), then execution commences at the statement following the case. Should none of the cases satisfy the expression then the case labelled default will be executed.
The default is optional. If it is not there, and none of the cases satisfy the expression then no action takes place. Naturally, cases must all be different, and they can occur in any order.
In our lesson example No 8, all that happens when a case has been satisfied (apart from the last - case '\n':), a message is printed to the screen stating which option from the menu was chosen. A more complete program would use each case to call an appropriate function for dealing with the option.
Perhaps you can figure out why the - case '\n': was included? Try writing the program with it excluded to see what may result.

```
/* a different example of switch
...case */
/* Functions are assumed to have
been designed in another part of the
program, together with any relevent
defination, etc */
switch(c)  {
 case 710:
    first();
    break;
 case 100:
    second();
    break;
 case 231:
    third();
    break;
 default:
    printf("  ERROR - CODE 31\n");
    break;
}
```

We have now studied the most powerful and most complex control statements in 'C', but we have one more to examine, the for ... command.

THE FOR THAT CAN BE A WHILE

The for... statement is very useful, but its job can just as easily be performed with a while... statement. The formal syntax for for... is:

FOR (EXPRESSION 1 ; EXPRESSION 2 ; EXPRESSION 3) STATEMENT;

This may appear complicated, but it really isn't.

1. The loop works by first evaluating expression 1
2. Test against expression 2
3. Cary out whatever is in statement
4. Execute expression 3
5. Adjust whatever is contained in expression 3, usually a loop counter
6. Test expression 2 and if true re-execute the loop until found false

```
/* Lesson 9 .. looping for . */
int l
 main()
 {
  for(l='a';l<='z';l=l+1)
  putchar(l);
 }
```

Here we have initialised l as an integer, and defined the main() function plus opening brace. Next comes the for...loop. It contains three" parts seperated by semi-colons

Part 1. l='a';

This is done once only, before the loop is properly entered.

Part 2. l<='z';

This part is the test or condition which controls the loop. If it is true, in other words if l is less than or equal to the character 'z', the body, or statement, of the loop - here putchar(l); is executed.

Part 3. l=l+1

This speaks for itself. Once done it loops back to the second part an re-evaluates the condition. The loop terminates when the condition is false, ie, l>'z'.

Let's examine putchar(l);

This is the command statement to output the variable letter onto the screen. It is essentially the output version of getchar() and is used in a similar way.

As with the while...statement, the body of the for...loop can be a single statement, as above, or a compound statement enclosed within braces. The choice between while... and for... is arbitary, based on what seems to be more clear. The for... is usually used in loops in which initialisation and re-initialisation are single statements logically related.

DELVING DEEPER - ANOTHER LOOK AT FUNCTIONS

As previously mentioned, a function is the equivalent to a subroutine in Basic or ML. the function provides us with a neat way to encapsulate some compulation or iteration within, what one might call, a 'black box'.

So far the only functions we have really deall with have been those which were included from the functions library, such as the getchar(), putchar() and printf() functions. Now it is time to write one of our own. Each function has the same format, and can be formally stated as thus:

NAME (ARGUMENTS IF ANY)
DECLARATION OF ANY ARGUMENTS
{
DECLARATIONS STATEMENTS
}

In the next lesson we will write a small function to raise the value of an integer by the power of 2. The function name is labelled power, and is driven by our usual main function.

/* Lesson 10 - a small power function */

```
main()  /* test power() */
{
    int c
     for(c=0; c<8; ++c)  /* ++c same
as c=c+1 more about this later */
                          printf("%d
%d\n",c,power(2,c));
}
power(x,n)   / * raise x to nth power
*/
int x,n;
{
    int c,p;
    p=1;
    for(c=1; c<=n; ++c)
      p=p*x;
    return(p);
}
```

As you can observe we have the main() function which, during execution, will call the function, power from within the line

printf("%d %d\n",c,power(2,c));

Each time the power function is called main will pass an argument to it, which, after computation, will return an integer ready to be printed. Notice also that the variable c in main, is unrelated to the variable c in power.

The value which the function power produces is returned to main by the statement return(p). Any expression may occur within the parenthesis of a return statement. It may also be stated that a return can be made without any expression, however, this has no useful value; so "falling off the end" by reaching the terminating right brace will do the job just as well.

CONSTANT DATA - AUTOMATIC, AND EXTERNAL VARIABLES

Any data item which is of a fixed value throughout the program operation is a constant. Other data can be variable and of several types. Two of these types are, Automatic and External. Another two types are Static and Register but of these we will discuss a little later in the series.

An Automatic, or as it is more commonly refered 'local', variable means that it is declared within a function, and is local to that function only. No other function may have access to it. This is an extremely useful facility which allows the programmer to use the same variable name in many different functions. Each local variable having no effect upon any other vcariable of the same name in different functions. Each local variable coming into existence only when ther function is called and to disappear once the function is exited.

Variables which are external to functions are commonly termed as 'global'. However, I find it best to reatin the use of the term external as we shall see in a moment. An external variable must be defined outside any function where storage is then allocated to it. The external

must be declared within each function which will require access to it.
Here is an example of both 'local' and 'external' variables from a section of a program which determines the longest line from a series of lines:

```
/* ..program section ..longest line .. */

#include<stdio.h>
#define MAXLINE 1000 /* maximum
                        input line size */
char line [MAXLINE]; /* external
                variable; the input line */
char save [MAXLINE[; /* external
                variable; the longest line*/
int max;            /* external variable;
                    length of longest line */
                        /* found so far */
main()
[
int len;                    /* local variable
              applicable to main only */
extern int max;     /* here the external
                    variable is being */
                    /* declared because
        main will require access to it */
extern char save[ ]; /* same as above.
        You will notice that the prefix */
                    /* extern is used. For
              this reason only I like to */
                    /* retain the use of
          external rather than global */
rop                 /* Rest of program */
```

You will notice that within the function which requires access to an external variable the keyward declaration extern has been used. There are circumstances where the extern declaration can be omitted. In fact the common practice is to place definations of all external variables at the begining of the source file, and then omit all extern declarations. More about this in later lessons. Please note also that I use the word 'defination' to mean the place where the variable is created and assigned storage, and 'declaration' to mean where the variable is stated but no storage space allocated.
External variables are initlallsed to zero by default, and local variables, for which there is no initialiser, are left undefined, in other words 'garbage' values.

DATA TYPES, OPERATORS AND EXPRESSIONS

Variables and constants are the data objects which the program manipulates.
1. Declarations list the variables, state what type they may be and can set their initial values.
2. Operators specify what action is to be done on them.
3. Expressions combine the variables and constants to produce new values for the workings of the program or functions.

VARIABLE NAMES

Names can be made up with letters and digits, however the first character must be a letter. The underscore "_" counts as a letter and is useful for linking long variable names together to make them more readable. ie; 'chars_test', 'first_data_list'. Only the first eight characters of an internal name are significant but more can be used for readability. There are some reserved keywords which cannot be used for varialbe names

```
continue register unsigned typedet
default   static   return   sizeof
switch    extern   double   struct
short     union    entry    float
break     while    long     else
goto      char     case     int
for       do
```

DATA TYPES

The data-types used by 'C' are;
int... float... char... short... long... double...
Short and long are qualifiers for int, which also uses unsigned, eg;

```
short intx;
long inty;
unsigned intz;
```

(The word Int can, and usually is, omitted)
The data type int means an integer number and float means floating point, or real, numbers. Char specifies a character data type, and long, short and double refer to long integers, short integers and double precision floating point numbers. An unsigned integer can only handle positive numbers. Check your compiler manual for the length of long and short integers, usually two bytes each for the CBM64.

MORE CONSTANTS

Scientific notation, such as 123.456E-7, or 0.12E3, is legal. Octal and Hexadecimal numbers are taken care of. A leading 0 on an int constant implies Octal, an 0x implies Hexadecimal.
A character constant is a single character within single quotes. 'x' 'c' 'a'. The value of the constant is the ASCII numeric value of the character. For example 'a' = 65: '1' = 49.
A string constant is a sequence of characters surrounded by double quotes as in;
"How long is a string?"
"" - a null string.
Certain non-graphic characters can be represented in character constants by escape sequences, such as;
\n - newling

\t - tabulation

\0 - null
Remember to distinguish between a character constant and a character string. 'x' = char constant but "x" = char string

MORE ABOUT DECLARATIONS

Variables must be declared before they are used, although certain declarations can be made implicitly by context The declaration specifies the type of variable, and then lists one or more variables of the same type.

```
int start,finish,step;
char a,b,name[50];
```

In the above two examples each type has three variables seperated by commas. They could just have easily been written;

```
int start;
int finish;
int step;
char a;
char b;
char name[50];
```

This takes up more room, but is useful for adding comments to each variable, or possible future

modification. Variables can also, with certain restrictions, be initialised when they are declared;
int c=0;
char letter ='a';
The following table gives some acceptable data declarations.

DECLA'TION	NAME	TYPE
char x;	x	character
char s;	s	character
int a;	a	integer
short b;	b	short integer
unsigned z;	z	unsigned int
char name[20]	name	char array up to 20 chars
int payoff[30]	payoff	integer array

ARITHMETIC OPERATORS

Arithmetic operators are, + - * / and the modulus operator is %
Integer division will truncate any fractional part. The expression x % y produces the remainder when x is divided by y - thus if y divides x exactly zero is the result. The % operator cannot be applied to float or double

RELATIONAL AND LOGICAL OPERATORS

The relational operators are; > >= < <= and the equality operators are; == !=
The unary negation operator ! (NOT), converts a non-zero or true operand into a 0, and a zero or false one int a 1. Common usage is constructions like; if (!name) - if Not name. The logical operators, && II (and/or). Expressions connected by && and II are always evaluated left to right and evaluation will stop as soon as the truth or falsehood of the result is determined.

INCREMENT AND DECREMENT OPERATORS

The operators are; ++ and —
The increment, ++ adds one to its operand, thus c++ would be the same as c=c+1. Likewise, the decrement c— subtracts one from c. The unusual aspect of the two operators is that they may be used as either prefix or postfix operators. For example; ++c increments c before using the value and c++ increments c after using the value. If c is 4 then x=c++ sets x to 4, whereas x = ++c sets x to 5, but in both cases c has been incremented to 5. The decrement operator works in the same way.

BITWISE LOGICAL OPERATORS.

These operators will only apply to integer numbers, not float, or double.
Bitwise AND..... .....&
Bitwise OR ....|
Bitwise exclusive OR.....^
Left shift..... ....<<
Right shift..... .....>>
Ones complement.....~

PRECEDENCE AND EVALUATION ORDER

OPERATOR	ASSOCIATIVITY
() [] -> .	left to right
! ~ ++ — - (type) * & sizeof	right to left
* / %	left to right
+ -	left to right
<< >>	left to right
< <= > >=	leftto right
== !=	left to right
&	left to right
^	left to right
|	left to right
&&	left to right
||	left to right
?:	right to left
= += -= etc	right to left
,	left to right

The operators -> and . are used to access members of structures. These will be discussed in later lessons, along with sizeof (size of an object), * (indirection) and & (address of). Note the operators, += and -=. An expression such as: c=c+2 can be written in the compressed form c+=2 etc.

TABLE OF BINARY OPERATORS

OPERATOR	ACTION
*	multiplication
/	division
%	modulus (remainder)
+	addition
-	subtraction
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
==	equal to
!=	not equal to
&&	logical and
||	logical or
?:	equivalent to if...then...else

ARRAYS

Arrays are allowed for all the data types. An array is a collection of data which all use the same name but subscripts identify the individual items within the array. In 'C' the primary subscript is always 0, and a three element array with the name data_list[3] would like this;
data_list[0] 1st element of array
data_list[1] 2nd element of array
data_list[2] 3rd element of array
Note the use of the square brackets to contain the subscript value of the array.
A string is an array. The elements of the string are single characters and the compiler will automatically place a 0 at the end of the string as a terminating marker so that
programs can easily locate the end. For example; char titles[5]

data element	**1**	**2**	**3**	**4**	**5**
array	**N**	**A**	**M**	**E**	**0**
subscript	**0**	**1**	**2**	**3**	**4**

IMPORTANT

A pointer is a variable which contains the address of another data item. 'C' has no direct commands for dealing with strings of characters, and this is where the pointer is extremely useful. It might, at first seem a disadvantage when compared with languages such as Basic, but not so because the programmer has a much tighter control over the data variables themselves.

Let us define a character array called titles.

```
char titles[50]; /* this decalres a 50
                    element character array */
```

Now we can declare a pointer to a character within the array.

```
char *cp;  /* by using *cp, cp becomes a pointer to a particular */
           /* data item within the array */
```

Next we can assign to the pointer cp, the address of the initial data item we wish to point to

```
cp=titles[0]  /* remember the first subscript in the array is always 0 */
/* which will point to the first element */
```

We will be looking into arrays and pointers more deeply as the lesson roll by. But to close this months tuition let us look at a function which will help us to understand more clearly arrays and pointers.

The function we will write is named fill() and would normally be called from the main program by the statement fill(c); where c is the address of a data buffer area (I shall be explaining more about that later in the series). The data buffer area, in this case, is presumed to be a character array of suitable size to hald all the data which will be typed from the keyboard. The initial value of c would be the address of the first subscript of the array.

```
/* .. lesson 11 .. an example of an array and pointer .. */

fill(c)
char *c;
{
    while((k=getchar()) !=eol && !=EOE)
    {
       *c=k;
       c++;
    }
    *c=NULL;
    return k;
}
```

The multiple expression line; while ((k....etc. Uses two expressions which I have not yet mentioned, namely, eol and EOE. This means that the variable k set to equal the value returned from the function getchar() will be tested for a carriage return - eol (end of line) and for a terminating key (in the case of SPINAKER'S POWER C a period '.' on a new line), EOE (end of eile). Some compilers might assign the run/stop key as an EOE marker.

I defined the cariable c to being a pointer to an element in the array by using the *c defination, and if one of the two end conditions - not equal to end of line and not equal to end of file, !=eol && != EOE have not been met then the loop statments are carried out, these are:

(a) *c=k; this will assign the variable k to the memory address which the pointer *c is pointing to, and,

(b) c++; then the pointer is incremented to point to the next memory location, ready for the next character. In this way the array is gradually filled as each character is typed from the keyboard until one of the two exit conditions is met. Once the exit condition has been met the line,

c=NULL;

will place an end of array marker 0 into the last element. The function then ends with a return to the calling function with the current value of variable k - return k;

SUMMARY OF THIS MONTHS TUITION

1. The introduction switch...case, used in conjunction with a series of logical decisions - such as in a menu.

2. The introduction of toupper which wrll evaluate a lowercase letter integer into an uppercase letter integer, tolower will do the opposite.

3. The introduction of compund expressions.

4. The use of default case, to trap errors etc.

5. The introduction of the for...loop - constructed in three parts and followed by a statment which can be single or compound.

6. The introduction of putchar() which has the opposite effect of gelchar().

7. How the for... is similar to the while...

8. We examined functions a little more closely, constructing one to raise values by the power of two, called from within a printf argument.

9. Constant, automatic and external variable data were examined where we saw that automatic, or local variable data is 'local' to the function which brings it into existence, and that the same data name can be used in any other function without clashing. We also discovered that external data, 'global', is defined once only and outside of a function, and can be used anywhere within the program.

10. We also considered data types, operators, expressions, and declarations, with an examination of arithmetic, relational, logical, and the bitwise logical, operators We saw the simplicity of the increment and decrement operators - demonstrating how an increment or decrement can be either prefix or postfix.

11. The inclusion of precedence and evaluation order tables and binary operator tables, for future reference.

12. A first look at arrays and pointers - discovering how a pointer (data address) will point to an element within an array, and finishing off with the construction of a function to fill a character array with input from the keyboard to demonstrate the principles of arrays and pointers.

I hope that this months tuturial isn't too long for you. 'C' is quite an extensive language and we need to come to terms with as much as possible each session. Ultimately, we can all benefit from the power of 'C'. Until next time...keep smiling.

FEATURE

THE PEEVES OF A SYSOP

STUART ALLEN

It takes a special kind of crazy person to make an, otherwise normal-looking, human being choose to operate a Bulletin Board system... Very often (s)he's placing valuable equipment at the mercy of others, and spending valuable time, and not really minding a bit. A little acknowledgement, the occasional 'thank you', and periodic uploads of fresh Public Domain software seem to keep these types happy. However (You knew there had to be a however to this, after reading the title, didn't you?), there are a few things that can... shall we say... P E E V E a system operator.

The bothersome users of a bulletin board fall into some broad classes, which are not necessarily mutually exclusive nor all that well-defined. Some are truly obnoxious, while others can get the SysOp upset with a third party, the world in general, or with... ummm... the SysOp.

Let us introduce the first, and most obnoxious, of those users capable of having an effect on the operator's blood pressure. We'll call him the 'Owner', because that's how the creature behaves... like he owns the BBS (not likely to endear himself/herself to the sap who pays the bills, no?). There are a broad range of symptoms displayed by an Owner. First of all, there is generally no respect given to all the other users of the system: (s)he'll log-on at 7pm and stay there downloading files until the system (or operator) hangs up on him/her. At other times, he'll call for a chat with the SysOp: if the SysOp's not there, (s)he'll leave a nasty message complaining about the SysOp's inattentiveness. If the SysOp is there, (s)he'll go on at lengths (typing slowly) asking why the files (S)HE wants aren't always there, and why the BBS isn't tailored to HIS/HER interests and needs. The only thing good about this creature is that (s)he will either

(1) get bored and suck some other SysOp's BBS dry, or

(2) get the message that (s)he's being a pig.

Next in our BBS Zoo we have the Bull. Owners tend to be Bulls, too. A bull behaves like (s)he's in a china shop, crashing about, spending tremendous amounts of time and effort discovering the variety of error messages the BBS keeps on file. Now, this sort of behaviour is not really frustrating to a SysOp... unless (s)he's waiting to use the BBS him/herself. What REALLY gets to a SysOp is that a full-scale bull will refuse to admit that it might be faster to read the help files, and if (s)he does have a question, (s)he'll pose it to the Sysop him/herself. This SysOp has a standard response: silence. Any question adequately answered in the Help files does not need further attention. Fortunately for the SysOps, most Bulls are self-curing. They will either discover the Help files or puzzle out the BBS structure by trial and error... and error... and error...

> **"LAST, BUT NOT LEAST, IS THE CLASSIC TWIT."**

Then there's the Apologist, who gets the SysOp mad at him/herself. What happens is the SysOp will complain publicly about some Owner or Bull on the system, and the Apologist will then leave a message saying (s)he's sorry for spending so much time downloading a file, or that (s)he tried various commands before going for Help. Why is the SysOp mad? Because a valued user has felt badly about the SysOp's complaints, while the REAL object of the whole mess has never read the rotten complaint, anyway!

Last, and probably Least, is the classic Twit. Twits are usually lacking in most of the social graces. A Twit usually is unable to spell (No offence, steve (SM30). I know it is your keyboard) nor comprehend 'courtesy'. He'll go right past the SysOp's request not to post messages offering to pirate software, and then post a message offering to pirate software. Requests for clean language are met with profanity, and when the offending messages are wiped from the BBS, the Twit will be absolutely outraged. Twits tend to run in groups, and will descend on a BBS like a hoard of locusts. Twits either grow up (mentally: a low physical age is not a prerequisite for one to be a Twit) or discover a BBS operated by a fellow Twit who tolerates them. The 'cracker' (what the press erroneously terms a 'hacker'), who delights in trying to crash remote systems to cause maximum discomfort and damage, is a true Twit.

Thank god there is still the Decent User. This is the person who will upload files someone's looking for, read the messages and pipe up if (s)he's got something to say, and will feel a twinge of guilt (needlessly) when he downloads that neat-looking big file or can't find anything

half-decent to contribute. They may not be thanked, but they're the ones responsible for the continued operation of a BBS. The thing is though, is any of this true? How many of you out there are bulls, owners, or twits? You never know, there might even be the odd Decent

"A CROSS BETWEEN OWNER AND DECENT USER."

Owner.

Talking of CNET (?), there are a few bulls/owners out there. I am like an owner, only not as bad.I keep downloading files, voting (Now. Listen up, all you peeps on the 'net'. I may be a scaled down version of an owner, BUT I AM NOT A '1' VOTER. If I ever vote 1 on a piece of software (god forbid), then I shall tell the creator why. I shall not go around aimlessly voting 1, 'just because I don't like the title'), sending (non-abusive) mail to people, etc. If you think of me as a cross between an Owner and a Decent User, then I don't think that you'll be far wrong.

THE PERILS OF SOCIALIZING WITH A MODEM.

THE TRAGEDY OF AN ON-LINE ADDICTION.

This article is based in America, where they use states instead of counties, so just bear with it, ok?

"Did you know that last month's flippin' phone bill was over $450? " my wife scolded me in her harshest, my-husband-the-child voice. "That's more than twice the monthly payment you make for that blasted computer!" she continued, as she escalated to screaming.

"I confess! I confess! " I sobbed. "I'm just an on-line junkie — I'm addicted to my modem! I guess I'll just have to join **My Modem and I**, breaking the habit of using it, before I owe everything that I have to the phone company. "

As a joint shareholder for **My Modem and I** breaking the habit of using it, referred to from now on as MMI-BHUI, I hear many different variations of that very same story every day. That awful disease, modem fever, is exacting a , tragically, large toll from the best, no, the very best and most useful people from our society's computer users. Modem- mania is sweeping through the very foundations of our country and there seems to be no way of stopping it! This disease (yes, it is a social disease of almost epidemic proportions) is becoming such a calamity that soon there's even going to be a soap opera about a father-son on-line addiction named, "I never liked them anyway."

If you don't already own one of these evil instruments of torture (to the phone bill), called a modem, then I warn you, nay plead upon you to never set your eyes upon one of these hideous things. At the very least, please, please don't buy one. Modem fever sets in very quietly; it sneaks up on you and grabs you by the ..... Yes, you've guessed it, it grabs you by the wallet, cheque book, or if you are that unfortunate, it could even get you by the CREDIT CARDS.

Once you own a modem (and I warned you about this, but would you listen??? NO!), you enter the insidious addictive trap by 'dialling up' (look, no hands! it does it all by itself. I suppose it is, in a way, an energy saving device) a friend who also has a modem (Didn't I warn you? You've even got your friend buying one). For some strange reason, typing messages to each other fascinates you both (Even if it is less than 10% of the speed that you can speak the same words over a normal voice link. There are, of course, the small problems that come with them, such as, never knowing what to say). Of course, you make several attempts of logging onto your friends computer, before realizing that at least one of you must be in the half-duplex mode; that discovery actually titillates you (sounds impossible, but true).

Just after you got over the shock of the half-duplex thing, your modem-buddy (friend is too good a term, after what they are about to do to you) sows another seed on the road to on-line addiction by giving you the number of a BBS (Bulletin Board Service). Once you get a BBS phone number, you've taken the first fatal step in a journey that can only end up with on-line addiction.

After you take the next step by dialling up the BBS your modem-buddy told you about, you find that it's very easy to 'log-on'. This weird form of conversation with an unattended computer is strangely exciting, much more so than just typing messages when you're on-line with your modem-buddy. The initial bulletins scroll by and inform you about the board, but you're too 'up' to comprehend most of it. Then you read some of the messages in the message section and maybe, in a tentative manner, you enter one or two of your own. That's fun, but the excitement is gradually beginning to wear off; you're calming down. Thinking that it might be worthwhile to go back and re-read the log-on

"THE EXCITEMENT NOW STARTS TO DROP."

bulletins, you return to the main BBS menu.

Then it happens. The BBS provides the bait that entices you all the way into the fiery hell of modem addiction. As you look at the BBS main menu to learn how to return to the log-on bulletins, you find an item called FILES. By asking your host

computer for FILES, you thread the bait onto the hook of corruption; the FILES SUBMENU sets the hook. You start running with the line when you LIST the files; you leap into the air with the sheer joy of the fight when all those public domain program titles and descriptions scroll by. They're FREE!!! All you have to do is tell the host computer to download (transmit) them to you. You download your first program, and you've landed, in the creel, cleaned and ready for the cooking fires (If you haven't noticed by now, it's describing the past experience, like a fishing trip). In just 55 minutes after you logged onto the board, you've downloaded six programs, one of them is Freddie Mercury's CBM-Talk, version four (Truly an instrument of evil).

BBLIST.DOC, which is also among the files you downloaded, contains a list of a great number of Bulletin Boards throughout the country (There's evil all around us, constantly tempting us!). You print the list and find about 60 BBS phone numbers (Have mercy upon our souls!). The list also gives you the hours of operation, communications parameters, and informs you about each boards speciality. You decide to try CBM-talk and use it to dial-up an BBS about three states away. Since the line is busy, you pass the time by entering the numbers into CBM-talks enormous dialling directory.

You try the number again — still busy. You think, 'Hey, there's one that specializes in PASCAL programs. Maybe I'll try it. It's about half way across the country, but it's after 5pm and the phone rates have changed. It won't be too expensive.'

The Pascal board answers. After 45 minutes you've downloaded another five programs. Then you call another board — only this one's completely across the country from California, in Dallas. And so it goes on into the night... And the next night... and the next...

Some days it gets to you. You begin to feel the dirtiness of modem addiction, particularly when your wife makes you feel like a child by berating you for those astronomical phone bills — if she hasn't divorced you by then. Every time you sit down before your Commodore 64/128 to do some work, play a game, do a bit of coding, etc., you dial up another BBS instead. If that one's busy, you call another, and another, until you connect. Then you feel OK, almost 'high' When you finally hang up, you still can't work you can only dial up another BBS.

Your downfall as an on-line addict is just another one of this society's terrible tragedies, such as polygamy or the compulsion to circle all the numbers on computer magazine 'bingo cards' or checking your 'lucky 7' numbers in the Sun. Eventually your whole social life relies upon only the messages you find on electronic Bulletin Boards; your only happiness is the programs you have downloaded (You never try any of them, only collect them).

"THE CURE IS SIMPLE. SET UP YOUR OWN BBS."

Still, hope exists. We, the dedicated, yet under-paid staff at MMY-BHUI, have done extensive research to find a cure for modem-mania, which has been ruining hundreds of lives, and we have succeeded in our quest. The cure is quite simple, but effective: Set up your own bulletin board service. Then all the other modem addicts will phone you, and their wifes can nag at them about $450 phone bills, and you can find peace — at last. Of course, the trouble with this, is that not many SysOps own a CBM 64/128 for their BB. I have also done vast research about the SysOps point of view. One, in particular, called Transylvania Express (the SysOp is Marcus Waterworth (told you I'd give you a mention)) is one of these helpful and informative boards, and put these questions to him.

1) What do you like/dislike about running a board ?
2) What hardware/software do you use ?
3) Roughly how much time do you spend on the Board ?
4) What types of people do you like on the board ?
5) Do you ever get the chance to use your computer for anything else. ?

The answers weren't really a surprise. He said, for the answer to number 1, "I really find it difficult to find many things I dislike about comms in general, as it gives you a chance to meet so many different people who you probably would never meet in any other way, and also it kills a lot of the prejudices people may have as they cannot see or hear the person that they are talking to"

This was a good answer to the question. It does give you an opportunity to meet so many different people who you would never have heard of, before. It also does kill of any prejudice between the SysOp and the User, or the User and other users You can't insult someone, if you cannot hear/see them. That is something that I don't like, anyway. People slagging off other people, for no just reason. If I went up to you, and started screaming blue murder about you, even if I had never seen you before. Wouldn't you think of that as unjust?? Yet, that is what many people in England (I can't say Britain, because I watched Points Of View once, and someone had written in about Wales and Scotland being blocked out, when they say 'and it's another victory to Britain' when it is a brit, and 'It's another victory for England' when it's a Welsh/Scottish person. So all you Welsh/Scottish people reading this article, I am not discriminating against you) think and

do. With comms, you cannot discriminate against other people, unless you prove why you said it.

For Question number 2, Marcus wrote (typed?) 'The board runs on a CPC6128 (Amstrad) with an 800k 5.25 second drive. I am planning on upgrading to a CP/M machine with a hard drive' I'll just break off, there. Normally, a hard drive is used, for the space, because you can fit much more onto a hard drive. He is also getting the second machine, so that he can do work on the board on one, whilst the board is running on the other. He needs to do this, for two reasons :

1) His board is 24hr, so can't get much time away from users, after about 5:30pm.
2) When he uses the second drive, he can only store 800k, so at the moment, he has to put up a g2 requests area, where people send him some mail, asking for a certain piece of s/ware, and it is put into the requests area for a few days. He then continued to say, 'as the only way I can give the callers the files they want at the moment is to have a requests area.', which is what I just told you

For the answer to number 3), he told me, 'I've never really thought about how much time I spend on the board, but it's quite a lot . If I'm not working or sleeping, I'm usually round the computer, somewhere'. In other words, Running your own BB can be quite time consuming, and if you can be bothered to put in the effort, then you'll get just as much out of it. Keeping a board spick and span can be quite hard, getting rid of any obscene language, clearing away the old text frames and sending mail out to answer problems, or annoying people asking him how much time he spends on the board.

Now then, Question four. This is the biggie. Is he pestered by any of the types of people mentioned in the other article? what does he do to them? These are both intriguing questions, so here are the answers. 'The only type of caller that I dislike is the one where the only thing they do and say is "My computer is better than yours" (Ah well, must stop saying that now), in one way or another. I know that this may well be true in some cases, but they don't seem to realize that there is a person at the other end of that computer, and that a computer is really just a tool to communicate, with the person behind that tool being much more important than any computer. There are a few callers that are 'difficult' or awkward that seem to think that the BB should just be for their use only. Well, let's face it... While they are here, they are leaving some other poor SysOp alone.' Looks like Marcus has got one or two owners of his own, to contend with Thankfully, most of his users are common, ordinary, run of the mill,

"SOME CALLERS NEVER EVEN READ THEIR MAIL."

decent users. They are the ones that keep the board going. I believe that he has had many more people join him on his board, ever since he went 24 hours. Before, it was 11:00 Pm to about 3:00 am. Not many people, especially schoolkids, postpersons, etc. Those people that have to get a good nights sleep, to be up early in the morning.

His answer to question 5 has been more-or-less answered in the previous paragraphs. For the time being, he can only use the computer for comms, until he gets the other CP/M machine. The software that he uses is BullDog, which was written by Blane Bramble and runs under CP/M. There is quite a lot of 'after sales support', so if you change machines Blane will sort out all the relevant details and overlays.'

Now for an outlook on life as a SysOp from a 'more seasoned' traveller This is from Bob Wilson (of Dabbers Lair fame). You can tell that he is more seasoned, because he writes more for the first answer:' The things I hate about running a BB are the fact that callers, especially new-users, don't take the trouble to read the information provided, then when they cannot make things go the way they should, they just drop carrier. Then there are the p******s (you should be able to guess the word. Just for a little help, try putting in the letters 'illock', somewhere in the word. They may be the right way around, or they might be mixed up, I'm not going to tell you!) who enter stupid names at log-on, although they never get in, I just zap them. There are the callers who never bother to read mail, either to them or public, they never send mail or reply to mail sent to them. Lastly, there are the rapists! (Not Jack the Ripper. We haven't told him the phone number of the board) Mostly they tend to be new to the board. They log-in, go straight to the files section, find they cannot access other areas and just drop carrier or log off, and never come back. There may be others, but these are the ones that really annoy me. All callers are welcome on the BB, the only ones I object to are the idiots who either don't know how to spell their own names, or those who can't read! Everyone else is welcome. The likes of running a board are chatting to callers, helping users with any problems, exchanging/providing files. There are the callers that will do anything to help, will phone up regularly. Some new callers even upload files on their first visit and send public mail and say thank-you to me for running the BB.' I told you he was seasoned, didn't I? (Didn't I tell you?). He has had more experience with running boards, unfortunately, he has also had more experience with the 'zoo' callers. In fact, there were one or two in there that I forgot to mention, but never mind. All's fair in love and

sub-nuclear atomic world domination, whilst burning the toast. (Well, it goes SOMETHING like that..)

Now then, after that speech, things are beginning to get a little bit smaller, as far as replies are concerned. He uses an Amstrad PCW 8256 (Which was modified by him to an 8512), with a Pace Linnet modem. He puts, in brackets, I could do with a hard drive, this would give faster access and greater storage space. So, yet another one without a Hard Drive. I must tell yah, fellas. Owning a 20 meg hard drive is lovely, almost like heaven on earth, but please, don't break out into a duet of the song, will yah? The question that I poised to him next, was a difficult one. He even said, the amount of time I spend, that's a difficult one. He spent a considerable amount of time finding out about software, and the running of BB's at the start. It took a very long time getting things set up, at first. According to the number of callers, and whether he had to backup the disks, which he does monthly, at the moment, it can vary from 4 hours, to 20 hours per week (Ah. I bet you thought I was going to say, per day, then, didn't you?). For the last one, he said, I do use my computer for my business, although I don't get as much chance as I used to, for other things. Sometimes it can be a nuisance, especially if you NEED to use the computer for yourself, and the BB is due online. The only way around that one is either to let the phone ring or preferably, to busy the phone (By taking it off the hook, etc.)

A few numbers for you to try (if you own a modem.)

Transylvania express:
Bradford (0274) 621668 (24 Hr.)
Dabbers Lair:
(0270) 624248 ***
The Owl Service:
Leeds (0532) 605876 (24 Hr)

Hours of operation

Days Online	From	To
Monday-thursday	20:00	00:00
Friday	20:00	01:00
Saturday	19:00	01:00
Sunday	14:00	00:00

Calling outside these times will mean that the call will be Voice Answered!!!

Unfortunately, I, have been Promoted to 'co-SysOp' on Transylvania Express. I am trying to get up quite a few things for the C64. It needs your co-operation, as well Please have a log-on, look around, leave a few messages, etc. Those who are only used to using Compunet will find that it is vastly different from the smaller Bulletin Boards. It needs some support from the CBM 64/128 owners in the region. If you do go onto it at all, then please leave me a message, to STUART ALLEN or STAPLE S. (It's ment to be one word, but you need to enter 2 (first and last name)), just to say that you log-onto the board.

C128 SPRITE EDITOR

PAUL TRAYNOR

At last, an easy to use sprite editor for C128 users everywhere

The C128 Sprite Editor will help you to create, edit, view and store sprites for use in your own programs. The program consists of two files 'sprite editor' (the main program) and 'sprite ed.mc' (the title screen). Also there is an example sprite file 'sp.music'

OPERATION

When loaded and run the program , will first display the title screen, hitting any key will give you the main menu, if no key is hit the main will appear after about 25 seconds. The options on the main menu are edit, view, disk and quit. to move along this horizontal menu you use the left and right cursor keys and press return to select option.

After selecting the Edit function you are given the option to alter the four sprite colours. Colours are changed by entering their respective numbers, as shown on screen, at each of the four prompts. After this you are taken into Sprite-Definition mode and at this point a help screen showing the commands available for Sprite-Definition mode will appear on the 80 column screen.

The View option, when selected, will again give you the option to change the sprite colours followed by a choice of viewing your sprites separately or animated, Separately means all 8 sprites will be shown on screen, Animated means that the sprites are displayed one at a time with the cycling speed chosen by you. Hitting any key will quit the view mode. When viewing, the sprites will always be seen expanded in both x and y directions.

The Disk function will present you with another menu, operated in a similar manner to the main menu, with options Load, Save, Directory and Quit. Each file of sprite data created by the Sprite Editor will be given a prefix sp. The Directory function will display all files from the chosen device number which contain this prefix. Loading and Saving will be done from the device chosen when obtaining a directory (default is 8) When entering a filename for loading or saving you do not need to enter the prefix. If you save a file with the same name as one which is already on disk the old file will be replaced.

Quitting the disk menu will send you back to the main menu as will loading or saving a file. When you quit the main menu you will be asked if you are sure before leaving the program.

YOUR OWN PROGRAMS

To use a file created by Sprite Editor in your own programs the command line is;

BLOAD"sp.filename",B0,P3584

Once loaded there are a host of BASIC commands and functions for using sprites, all explained in the System Guide. Below is just a summary;

BASIC Commands

SPRITE; sets up parameters for a sprite, ie. on/off, colour, screen priority, x and y expand and mode.

MOVSPR; used to move or position a sprite on screen.

SPRCOLOR; sets the two multicolours for sprites.

SPRSAV; used to transfer data between sprites or between sprites and text string variables.

COLLISION; sets line number to move to when a collision is detected.

BASIC Functions

RSPRITE; returns sprite characteristics set by SPRITE command.

BUMP; returns sprite collision information.

RSPPOS; returns the position or speed of specified sprite.

ADVENTURE HELPLINE

JASON FINCH CONTINUES WITH HIS JOURNEY THROUGH KRON

Here we are again with another edition of Adventure Helpline, covering the adventure KRON published by CDU in December 1989. Last month I ploughed through nearly half of the locations and in this issue a further seventeen are covered. For the format of the information below please consult last month's issue. At the end you will find a conclusion section to round up a few things to be said in general about stage two, and because the complex system of caves are found in this stage, each of the locations depicting caves contains information that will allow you to return to the first cave so that you can start again if you get lost. So first of all let's just repeat the last location of stage one, where I left you last month.

23

On the north shore of Sark north of the great Caves of Goth, an underground maze of tunnels and pits. Somewhere in the depths of this subterranean world is the Cave of Ice said to hold the key to freedom.
Exits: NORTH 19, SOUTH 24

24

Inside a tunnel leading to the caves. More tunnels lead north and east.
Exits: NORTH 23, EAST 25
Type: RUB TWIGS - they are set alight
LIGHT BRANCH - the branch burns and gives off a glow

Other info: When you go east to location 25 you will need some light because caves are generally dark. This is provided by the twigs and branch, obtained as detailed above.

25

Deep in the caves of Goth. Tunnels lead in all directions.
Exits: NORTH 25 (no effect), EAST 26, SOUTH 27, WEST 24

Other info: Here you should go east if you have not got the nugget yet. If you do have the nugget then go south.

26

In the Silver Mine. Tunnels lead in all directions. Lying in the dust is a skeleton holding a rotted spade. Large rats, their eyes blinking in the sudden glare from your torch, scurry past.
Exits: NORTH 23, EAST 25, SOUTH 24, WEST 25
Type: EXAMINE SKELETON - the finger points to a recess in the wall
EXAMINE RECESS - you find a nugget
GET NUGGET
EXAMINE NUGGET - on it is written the word "Okura"

Other info: The spade is a red herring (I think!) and will simply crumble into the dust if you try to pick it up. To return to location 25, enter W.

27

In the Caves of Goth. Tunnels lead in all directions.
Exits: NORTH 26, EAST 28, SOUTH 30, WEST 27 (no effect)

Other Info: Unless you have not recovered the nugget from location 26, go south. To return to location 25, enter N,W.

28

In the Caves of Goth. Tunnels lead off in all directions.
Exits: NORTH 28 (no effect), EAST 28 (no effect), SOUTH 28 (no effect), WEST 27

Other info: To return to location 25, enter W,N,W.

29

Inside a tunnel west of a cave is the cage of the Albino Man. The door to the cage is open.
Exits: EAST 30
Type: GET STONE
EXAMINE STONE - on it are the words "Omar Kabul"

Other info: I have still not found out the significance of the Albino Man's cage being open. I don't think it is anything to worry about! To return to location 25, enter E,N,N,W.

30

In a large cave by a stagnant pool. Its water is deep and motionless. Tunnels are north, east and west.
Exits: NORTH 27, EAST 31, WEST 29

Other info: Do not type JUMP or try to get into the pool because you will die and the adventure will therefore be over! Also, do not move east unless you have BOTH the stone from location 29 and the nugget from location 26.

31

In the Cave of Ice. Ice and snow cover the roof and the walls. To the east over a narrow but deep pit is a passage. A silver door is set in the south wall. West is a tunnel.
Exits: WEST 30, JUMP 32

Other Info: When you enter this location your branch will go out automatically preventing you from re-entering the darker caves to the west without being killed. Therefore do not enter this location unless you have collected the stone and the nugget from within the caves. Also do not type EAST as you will fall into the pit and die. To cross it enter JUMP.

32

In a passage. To the west over a narrow but deep pit is a tunnel. There is daylight above.
Exits: CLIMB 33, JUMP 31
Type: RUB LAMP - a genie appears and tells you to utter the words in gold. These are the words written on the golden stone.
OMAR KABUL - this should then be typed as if OMAR were the verb and KABUL the noun. The genie will recite a little poem for you!: "He who has the power still, Lives alone upon a hill. Give to him the stone so bright, Then kneel and he will help your plight."

33

On the main shore of Sark. To the south of the Valley of the Stones. A rough track leads upwards. A secret tunnel leads down to the caves.
Exits: NORTH 20 (in first stage), DOWN 32, CLIMB 34

Other info: From here you should climb upwards until you reach the old Boran hut.

34

At the top of a rocky incline next to a hill. From here the Sea of Storms looks quiet and peaceful. To the south is a deep chasm.
Exits: SOUTH 35, DOWN 33, CLIMB 36

Other info: Do not go south - if you do then you will fall into the deep chasm and the adventure will be over.

35

You fall into a deep chasm...
Exits: none

Other info: As the description says, you fall into a deep chasm. You die and the adventure is over. Do not enter this location.

36

On a hill outside the door to an old Boran hut. Smoke rises from a rough stone chimney. Paths lead west.
Exits: WEST 38, DOWN 34
Type: TAP DOOR - an old man opens the door
OFFER STONE
KNEEL - the man invites you into the hut, location number 37.

Other info: Once the man has opened the door, don't hang about. Get straight into offering him the stone and kneeling down. Otherwise he will shut the door and you won't have got anywhere!

37

Inside the old Boran hut. There is a door to the west. A log fire burns brightly, its friendly glow lighting the crude but comfortable dwelling.
Exits: WEST 36
Type: OFFER SCROLL - the man will decipher it and then give you a shiny shield. The man reads: "When walls may move your senses so, try south then east then south again if on this earth you would remain."

38

On open ground, there is a rough track to the west and a deep chasm lies north. There is a hut to the east.
Exits: NORTH 35, EAST 36, WEST 39

Other info: Do not go north unless you want to be fall into the chasm and die.

39

On a rough east to west track. The moon vanishes behind the clouds. You begin to feel uneasy as though something or someone is above you!
Exits: EAST 38, WEST 40

Other info: Don't worry about being watched - just go west to the final location in this stage.

40

In the Valley of the Dead. Due south over a deep lake stands the Castle of Spells, its stout walls rising towards the sky. Huge battlements overlook the lake and arched windows like cold dark eyes stare down into the valley.
Exits: EAST 39 (I think that is possible!)
Type: PLAY FLUTE - the eagle swoops and takes you over to the battlements of the castle (next month's article begins there).

Other info: If you attempt to negotiate the lake by going south then you will die instantly. The lake is deep and icy which you just can't cope with!

I hope you can now finish that section so that we can start in the castle next month. The locations that you should visit, in the "correct" order are as follows: 23-24-25-26-25-27-30-29-30-31-32-33-34-36-37-36-38-39-40

You must get both the stone and the nugget so that you know what to say to the genies and the old man, and how to get through a door that crops up in the next stage. Only throw things away once you have used them. For the next stage you will theoretically only need the shield from the Boran man and your magic ring. You can take the nugget as well if you like. In the system of caves, it is quite easy to find your way around. Locations 26, 29 and 30 are easily identifiable by things that are permanently there, and location 25 is the only one to say that you are "deep" in the caves of Goth. The other two are locations 27 and 28. If you think you are in one of these just type EAST. If in 27 this will take you to 28 and if already in 28 you will remain there. You therefore then know that you are in location number 28. But don't trek around the caves for too long because your branch will burn out (I think you are allowed approximately fifteen moves - but don't quote me on that!).

That wraps it up for this issue. See you all again next month when the third and final stage, consisting of an unlucky thirteen locations, will be discussed. Strictly speaking there are fourteen, but one is the final location where it all ends so I'm not counting that! After that I think we shall move on to another adventure. That being "The Astrodus Affair" written by Mark Turner and published in the June 1990 issue. So until next month, happy adventuring!

TECHNO-INFO

Problems, problems, problems...everybody's got a problem. So let Jason solve them for you

JASON FINCH

Dear CDU,
Since the fifth issue of volume one I have been a subscriber to CDU. Two years ago I decided to buy a 128. Here in Holland it is almost impossible to find any software for this machine. Luckily CDU has been providing me with some excellent programs for the 128. Please continue with it. On the June issue's disk there were some more programs available for the 128. But not for me... The whole directory consists of "unscratchable" files, the ones with the less than sign behind the filetypes. When I tried to load any of the 128 files the computer responded with a file not found error, although the programs do load when I'm in 64 mode. I've spent nights searching for a possible solution but have failed. Could you please help me.
Marc De Loor, Holland.

Dear CDU,
With reference to your letters from D.Taylor and W.Nisbet in the August publication. If they lock the drive of the 1571 into the 1541 mode with the command OPEN 1,8,15,"U0>M0":CLOSE 1 before loading, I think they will find that this will overcome the problem of loading the POPP and Converter/Maths Aid programs. Also, congratulations on those two excellent 128 programs.
Norman Linney, Kent.

Dear Marc and Norman,
Thanks very much for both of your letters. You are right Norman, to solve the problem the command you gave will work and in both the past two issues I have explained the cause of this problem and how to rectify it permanently, although unlike you I have not provided any practical information to actually solve it - until today that is! Marc, your letter arrived before they were published. Perhaps by now you have been able to solve the problem yourself which you correctly identify as the fact that the files are "unscratchable". The programs load in 64 mode only because then your drive, which I blindly assume is a 1571, is in 1541 mode. If you switch to 1541 mode in C128 mode then they should load. This is exactly what Norman's command does. To clear the problem once and for all you will find a program on the disk, filed as "PROB1", that will ask for a filename, hunt for it on a disk in device eight and then render it "scratchable". Before hunting for it (the filename on the directory track) the program will ask if you are aware on which sector it can find the information for that particular file. Usually you won't, so just press "N". I have programmed this to operate in 64 mode in case this issue's disk also has the "unscratchable" files. This program should then allow the CDU programs to load in C128/1571 format.

Dear CDU,
Could you please explain how to load in the screens created with "The Image System" as the manual unfortunately leaves a lot to be desired.
Mr A.Cowan, Scotland.

Dear Mr.Cowan,
The version of the Image System that I own does have a manual that could be better although on page eleven there is a technical section and on the last page, page twelve, there is a program listing that will

enable you to load the pictures and display them. In case this is different to the manual which you own, I have included a program of my own, written in BASIC, that can be found on this issue's disk under the guise of a file called "PROB2". I hope you find it of some help.

Dear CDU,
I am writing to you hoping you can help me with this problem. I have made telephone calls to various manufacturers of Commodore 64 software in an effort to obtain a game of Bridge for the 64 (disk or cassette). Unfortunately nothing turned up. I am now housebound and cannot find any other Bridge players. I find playing the computer satisfactory as I had a copy of Bridge for the MSX. Have you any suggestions on how I could get hold of a copy?
Mr Wright, Shropshire.

Dear Mr.Wright,
I am unaware of a commercially available game of Bridge but obviously I do not have a knowledge of every piece of software produced although I would hazard a reasonable guess that somewhere out there lies a game of Bridge compatible with the 64. Therefore the only thing I can do is to ask if any of our readers knows from where such a game could be obtained. Or perhaps someone has got a game of Bridge for sale. If any of you can help then please write to us with any relevant details

Dear CDU,
My first query is that I have a 128 with a 1541 disk drive and a 1701 monitor. I read in May's edition the letter regarding the use of the nine pin socket at the back of the 128. Would it be possible to couple from this into the 1701 monitor to get an 80 column screen? My second query is that where, if possible, could I get a print head for an MPS801 printer. I have tried quite a few places up to now without any joy. I am an ex-miner disabled now and I spend many hours on the computer but I am stuck now that my printer has broken. I have got the use of a Citizen 120D for a few days from a relative but he now requires it back.
K.Taylor, Barnsley.

Dear Mr.Taylor,
Unfortunately the 1701 does not have an RGB port and therefore you cannot connect it directly to the RGB port on the back of the 128. However, do not despair! A company called FSSL stock an item called a 40/80 column adaptor which will work with all monitors that have a composite video connection, which the 1701 does. This adaptor will allow an 80 column display from the 128 with your monitor. Their address is FSSL, Masons Ryde, Defford Road, Pershore, Worcestershire, WR10 1AZ and their telephone number is 0386-553153. The product costs five pence short of twenty pounds (time to try out your maths skills!). With regard to your second query there is a firm in Birmingam called HRS Electronics Plc that stocks MPS801 print heads. Their telephone number for sales is 021-789-7575. Although I am not sure, I think the print head will set you back about fifty-three pounds! I know it's a lot of money, anyway.

Dear CDU,
I am interested in the Commodore 1764 RAM Expansion Unit. Just a couple of quick queries. Firstly, can I play games with it plugged in? And secondly, can I use cartridges with it plugged in?
Ludwig Flask, Malta.

Dear Ludwig,
To answer your second query first, very few cartridges, if any, can be used when the REU is in place (of course having said that, I will receive dozens of lists of things that can be used). Secondly, all other software should work perfectly well when it is plugged in.

Dear CDU,
I have been waiting for the past three months for any indication that my game (Fast Future) is to appear in CDU. Can you please inform me when the program will be published.
Paul Black, Scotland.

Dear Paul,
Unfortunately it is not possible to tell contributors like yourself exactly when a program will be published. The procedure is usually as follows. Firstly an acknowledgement is sent to you, followed by either an acceptance form (agreement form) or a letter saying that your program, for one reason or another, has been rejected. This can be, but rarely is, up to two to three months later. From then you must just sit and wait. We have floods of contributions and a balance of software in each issue is what is aimed for. Therefore it is not possible to plan far in advance exactly what will appear in the next, say, six issues. I am afraid you will just have to be patient. But if you have not even had an acknowledgement letter yet then please contact Paul Eves - I am sure he will then offer you some help.

Dear CDU,
I use my trusty 64 for my amateur radio hobby and chat to others in a data mode from the keyboard and another 64 is used for word-processing and using the CDU programs. It is a great pity that I do not have expertise on the 64 in this area of programming for this computer. I would certainly like to enrol as a night class student even though retired, but as far as programming is concerned I am a duffer. This leads me to my main question! I have a BASIC program

LETTERS

here which I very simply converted from an 'address book' type program to 'log book' use, which is required by amateur radio operators to keep track of ALL we speak to. Unfortunately it was written by a person who is sadly no longer with us and it is not quite right or I should say not working the way I would like it to. Can you please recommend a person who is capable of perhaps getting it to work properly, perhaps put it into machine code at the same time and then it can be released in some way. I hope you can suggest the right person and I will gladly send them a disk and a hard copy with my comments how I would like it to work. Thank you very much.
Ken Mottley, Wales.

Dear Ken,
Thanks very much for your letter. You ask if I am aware of anyone who would like to undertake the task. Well I am stepping forward myself as a possible candidate. If you could kindly send me, at the address given at the end, all the items that you stated then I shall have a look and see what I can do for you.

Dear CDU,
I am writing in the hope that you can supply me with an address that I can use to obtain an instruction manual for use with the graphics software "Mouse and Cheese" by Neos. The software came with the second-hand C64 that I bought and I would really like to obtain the instructions. I hope you can help in some way.
J/T P.H.Clement, The British Forces.

Dear Mr.Clement,
The only address that I have for Neos is: NEOS Europe, 26 Wycombe Road, London. However I am not sure how up-to-date this is. The instructions for Mouse and Cheese are not really, in my opinion, necessary because everything becomes quite self-explanatory after playing about with it for a while. The instructions don't help too much - it is only two sheets of paper - because of the excellent way in which the program has been designed.

Dear CDU,
I have now entered the wonderful world of disks and I am transferring my tapes to disk but unfortunately I cannot save a multiload game in full - only in blocks which after completing one level I must reset and then load the next block to continue to the game. This shows the same score and so on each time. Is there a way to save and complete the whole game straightaway?
Ronald Gushlow, Croydon.

Dear Ronald,
There are a number of plug-in backup cartridges available that transfer multiload tape games to disk. They are still saved in blocks but each is loaded automatically as if the program was on tape. The game can then be played continuously. However it is very difficult for me to recommend a cartridge as most may require what are called 'parameters' of one sort of another to make the necessary changes to specific multiload games. Unfortunately you did not supply the name of the game you are having problems with. I wish you luck in finding a solution.

Dear CDU,
I don't know if you can help me, but I recently found a so-called "cheat code" for the game Operation Wolf. Great, I thought, and immediately typed it in following all instructions given. However all the program seems to do is to print a message informing me that I have entered the data incorrectly and nothing else. I have enclosed the code listing in the hope that you can shed some light on this.
Mike Pitches, Plymouth.

Dear Mike,
A bit of a specialised one this, but in case anyone else has the same program and it is the published listing at fault - which I don't know - I have decided to incorporate your query in the magazine. So a quick answer. The error is in line number 60. You must change the value 53280 to 53230. Hopefully that should do the trick.

Dear CDU,
Can you please help me. I bought a "Manager's Special Offer" from my local branch of Dixons - it is a Serial 8056 Thermal Printer for just over ten pounds. Yes! - ten pounds. There are no instructions but printed on the box is "80 column width, 56 cps, RS232 serial interface". The printer lead has a rectangular plug with 16 holes - eight along the top and eight along the bottom. Can you please advise me on what I need to get it working on my Commodore 64. Dixons could not tell me and I have written to a couple of firms but despite SAEs they never gave me the courtesy of a reply. Have I bought a pig in a poke, albeit a cheap one, and if not could you advise me of a firm that can supply the lead or interface or do I have to put my "bargain buy" in the bin?
James Sloan, Scotland.

Dear James,
I don't think you need to put your printer in the bin just yet. First of all try out a company in Manchester called Meedmore Limited. I am not sure of their full address but the telephone number is 061-251-2202. I have been in touch with them and it certainly sounds as if they can help you. What you need is an interface to convert from a 64 port to RS232. Their product for this purpose is the Stack RS232 Interface and costs just short of thirty pounds. Then you will need an additional cable. The RS232 interface comes with details of how to make your own but if you don't fancy that, for an extra thirteen pounds they will

make you one up, so long as you provide as much information about the pin configurations as possible. I hope I have been of some help

Dear CDU,
With reference to a letter in August's magazine concerning the MPS801 printer for sale. After reading that, it came to my notice that I have got a printer of the same make and that when I print certain letters, such as P, Q and Y, in lower case, they are all positioned above the normal line of the other letters. I am interested in finding out if the chip concerned is still available.
I.A.Sutherland, Wisbech.

Dear Mr.Sutherland,
The printer in question is fitted with the Printer IV chip that was on the market just over a year or so ago However, that particular chip (or for that matter any hardware producing similar results) is no longer available, at least in Britain. I say that not because I know of somewhere abroad that still sells the product, but just in case somewhere abroad does sell it.

Dear CDU,
I possess a Commodore MPS801 printer which has the annoying feature of not having true descenders. Do you know of any hardware that would resolve the problem? As I use my printer mainly for letters, would a software package, such as GEOS, solve the problem? Thanking you in anticipation.
David Whittaker, Cheshire.

Dear David,
Yours is a similar query to the previous one and as said in reply to that, unfortunately there is no such hardware available in Britain today. But the software package that you mention, GEOS, would solve the problem This excellent package gives you much more versatility than you would normally have with the MPS801. You can choose from a multitude of different fonts, and can also obtain bold, italics, underlined text and so on. All these, because the characters are sent to the printer in a totally different way to normal, will have true descenders. This package is available exclusively from FSSL. Their address and telephone number can he found in my reply to the fourth (or titth - depending on how you view the first two!) query, from Mr.Taylor.

Dear CDU,
Firstly, how do I obtain back issues of CDU from the very first issue up to Volume 2 Number 4? Second, if I obtain a copy of CDU which is slightly bent and find that I can't run any programs or files from this, is it possible to copy the CDU disk onto a blank disk and then run the programs that way? If I can't copy a disk which is slightly bent onto a new disk then could I send the disk back for a replacement? If so how much will it cost and what will the disk be sent back in?
S.Morgans, Kent.

Dear Mr.Morgans,
Back issues of CDU that do not appear in the "Back Issues" section at the front of the magazine used to be able to be obtained from the publisher of CDU when a special coupon came along in the magazine. However, since the magazine has changed publishers I am not sure what the arrangements are. Perhaps you could give Alphavite a buzz on 0908-569819. I am sure, though, that any back issues beyond those shown in the section at the front of CDU will be photocopies of only the pages required for use with the programs on the disk. If your entire disk is faulty you will not be able to transfer the files to a new disk and get them to work. Only if any of the original programs work will you then be able to load them and then save them to the new disk and those ones will work. Remember this is only if the original files were uncorrupted. Details of how to send disks back and get replacements can be found at the end of the "Editor's Comment" section at the start of the magazine.

Dear CDU,
Just a couple of quick queries. C128 Windows, published in Volume 1 Number 4 of CDU produces a blank screen when I run it. I have a C128, 1571 disk drive, 12" black and white television and no add-ons. Secondly, can you tell me if the first article of the 65xx Interfacing series by Steve Carrie was published in the April, May or June issue of CDU, as I was in Australia at the time.
D.Callaway, Salisbury.

Dear Mr.Callaway,
The reason for the blank screen is that C128 Windows runs in the 80 column mode of the 128. With a standard television set it is only possible to obtain a 40 column display Therefore no display is produced. The 65xx Interfacing series actually began in the March 1990 issue of CDU.

Tip of the Month

Alas there have been no more envelopes falling through the letter box containing details of tips from readers. So this month I shall pick out a bit of knowledge that may he of use to all BASIC programmers. Many people deprive themselves of the wonderful scrolling messages when programming in BASIC because they are unaware of how simple it is to achieve. Ok - it is a rough scroll and not as impressive as a smooth one operating under machine code control. But I do not see why that should worry anyone. Here is what you

should do. First of all define a string that contains your message remembering to put a space between the last letter and the quote marks: 10 M$="TECHNO INFO ". Then what you should do is use the following line to move the message (which can be up to 254 characters): 20 FOR A=1 TO LEN(M$) STEP 0.2: PRINT "(crsr up)";MID$(M$,A,39): NEXT - If you should want to change the speed then alter the value 0.2 to something else. I hope someone finds that helpful.

But wait, what is this? A letter arrived this morning with some tips in - and some very worthwhile ones I may add. They come to you courtesy of Mark Carroll in Cornwall. Firstly, if you have a file on a disk that contains important data, but you forgot to close it, if you do: OPEN 1,8,3,"filename,P,M" then you can read it like a normal file without Write File Open messages. Secondly, POKE 808,251 will disable RUN/STOP and POKE 808,237 enables it again. In machine code, the commands LDA #lo-byte, LDY #hi-byte, JSR $A81E can be issued to print a string terminating with a $00 byte. It must not be longer than 255 characters. Fourthly, JSR $E716 instead of JSR $FFD2 will print to the screen even if the output is directed elsewhere, such as to a printer. Finally, in the C128's 64 mode, POKE 53296,1 goes into 2MHz and POKE 53296,0 reverts back to 1MHz. This has no effect on the normal C64. Thanks Mark for those - hopefully some more people will send some in and we can continue the run of being able to print tips from the readers.

Unfortunately though that rounds off this issue's Techno Info. If you are experiencing any problems in operating the CDU programs (please note the subtle difference between that and the disk being corrupt) or if you have any programming queries then please write to the following address: CDU Techno Info, 11 Cook Close, Brownsover, Rugby, Warwickshire, CV21 1NG. That is also the address to which you should send anything that you wish to be published in this Tip of the Month section. Incidentally, thanks to everyone who wrote in saying that they would buy my (alias Jason Finch - cunning bit of free advertising, don't you think?!) MPS801 printer. It has now been sold. See you all again in the November issue!

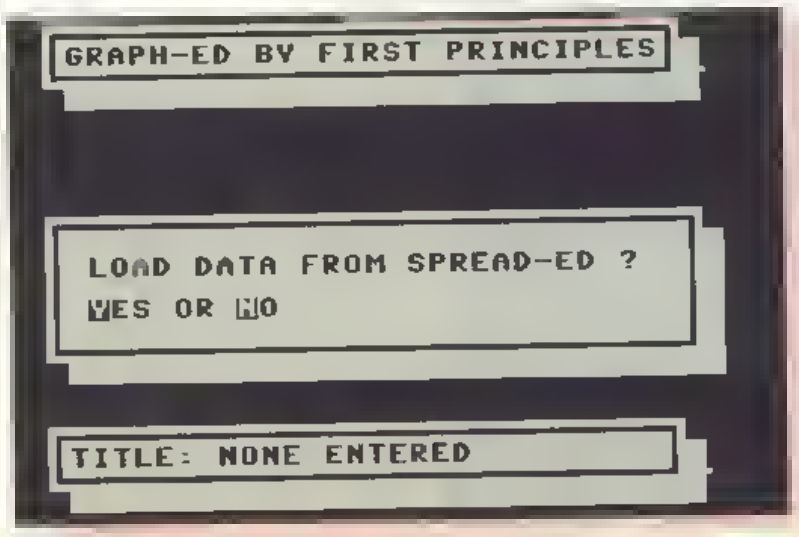
GRAPH-ED

A Graphs Editor utility that compliments SPREAD-ED found on page 9 in this issue

FERGAL MOANE

A picture is worth a thousand words and this is especially true of numbers. A graph or chart can tell you at a glance what would take a column of figures minutes to do. Spreadsheet columns and rows tell you very little but a graph is much more enlightening. A graph can show trends and predictions that rows of numbers cannot. With this in mind, I designed a stand-alone graphing utility to complement the spreadsheet program, SPREAD-ED, found on page 9 in this issue. It can import labels and figures from SPREAD-ED but also provides full input facilities for you to create independent graphs. The printer routines make the best use of the MPS801/803 printers and some very professional results are possible. GRAPH-ED is not a presentation tool so there is little point in using colours or shading as the printed results are often poor. I have opted for a plain black on white display which gives the user an idea what the graph will look like on paper and also gives clear output.

USING GRAPH-ED



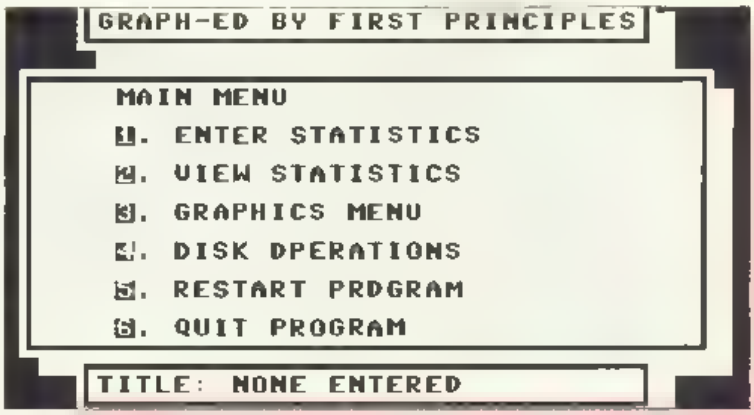
I have tried to make GRAPH-ED as user-friendly as possible, incorporating the same pseudo-windows and requesters that are used in SPREAD-ED. However, no program is completely idiot-proof and you should avoid using excessive values in response to input requests. GRAPH-ED will never crash, (Never say never...ED!!), although it may return to the startup-screen, losing all your data. Save your data regulary to avoid mishaps. I have included a memory map of GRAPH-ED for interest sake. Note that you can load your own character set into 49152 ($C000) and the label routine will use this new character set.

0-1024 ($0000-$0400)	Basic workspace
1024-2048 ($0400-$0800)	Screen
2101-16384 ($0835-$4000)	GRAPH-ED machine code
23552-24552 ($5C00-$5FE8)	Hires colour
24576-32768 ($6000-$8000)	Hires screen
49152-51200 ($C000-$C800)	Character set
51200-53248 ($C800-$D000)	Hires routines

The best way of describing the GRAPH-ED functions is to detail each menu option seperately. This is done below.

LOADING SPREADSHEET FILES

Before the main menu is presented, you are asked if you want to load data from SPREAD-ED. If you do, you should have a file saved from within SPREAD-ED (distinguished by it's .SHT extension). You should proceed and enter the filename (WITHOUT the .SHT extension) and GRAPH-ED will attempt to locate the file. Note that GRAPH-ED can only handle at most 16*16 spreadsheets. This is uaually more than adequate, but you may have to edit your sheet in SPREAD-ED before loading. Then choose the row number. GRAPH-ED will then import the labels and numbers in that row. Note that the inclusion of formulae in the sheet will not crash the program but are not evaluated correctly.



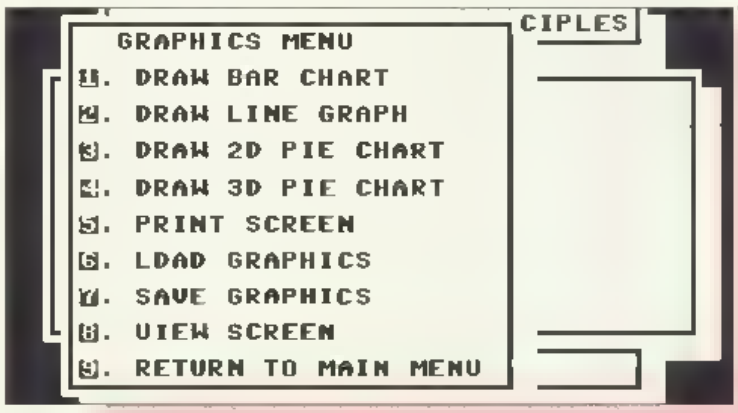
MENU OPTIONS

ENTER STATISTICS

This is the option that you should choose if you are not importing data from SPREAD-ED. You will be prompted to enter the title for your graphs. This is the title that will be displayed at the top centre of each graph and it can be up to 32 characters long. Press RETURN to keep any title that may have been previously entered. The Input screen will then be constructed. It consists of 16 units with a label and value in each. Labels can be up to 10 characters long and values van be 8 figures 16 units is a sensible limit as the graph segments become too small to be distinguishable after this limit.

To enter your data, enter a label followed by RETURN and then a value

followed by RETURN. The delete key is operational to correct mistakes and any value currently in the unit may be preserved by pressing RETURN Once you are in the label entry section (shown by the white cursor), you have the option of moving forward or backwards in your entries. Pressing the up and down cursor keys will move you back and forward to correct mistakes. You may also exit now by holding down CTRL. Note that you may only exit when you are entering a label and while entering a value. Although ten characters are available for labels, I would advise against using more than four characters as the labels tend to take up a lot of room and are knocked out of position. Three letter lables are ideal (JAN, EEB etc). There is no real limit on numbers but with any graph you should try and ensure that the numbers are around the same value or the scale may have



difficulty representing them accurately.

VIEW STATISTICS
This will display all labels and values currently resident in GRAPH-ED. It is useful to check that the values are correct after loading Spreadsheet data. You also have the option of dumping the data to your printer. Press 'Y' when prompted and make sure that your printer is on.

GRAPHICS MENU
This menu is the heart of the program and is where all graphics related functions can be found. The graphs themselves are drawn quite quickly but the labels are very slow. Although the routines are in Machine Code, some very complex and therefore slow calculations are needed to transfer the characters to the screen. The advantages are that the graphs are WYSIWYG (What you see is what you get) and the output is better than using the printers standard fonts. Also, you can load in any character set into the 2Kilobytes starting at 49152 ($C000) and customise the text to your own tastes. If the long delays begin to annoy you, you can quit, as always, by holding down the CTRL key then press any key to return to the menu.

DRAW BAR CHART
A chart consisting of a number of vertical filled bars is constructed to scale. As in all graphs, there is no limit on the size of numbers as all have a fairly 'intelligent' scaling routine. The bars become thicker and thinner depending on the number of units of data entered, so the lengths of labels may need to be modified as the bars become thinner. Once the labels have been copied to the screen (be patient) press a key to return to the menu.

DRAW LINE GRAPH
Similar to the Bar Chart except it consists of straight lines joining the points.

DRAW 2D PIE CHART
Draws a conventional pie chart and labels the segments in the appropriate places.

DRAW 3D PIE CHART
Draws an elliptical chart which gives the impression of three dimensions. The formula for computing the positions of the labels is not suited to ellipses and the labels may sometimes appear slightly out of position. To compensate for this would have made it too slow to be useable.

PRINT SCREEN
Performs a dump of the current screen to the printer. It is designed for a normal device 4 Commodore printer (MPS801/803) although I have used it with my own Star LC10-C with good results. The dump is centered on the page and a print buffer results in quite fast output.

LOAD GRAPHICS
Loads a Hires graphics screen back into the GRAPH-ED screen area. This will load a file with a .GRA extension but do not type this extension into the filename.

SAVE GRAPHICS
Saves the current screen with a .GRA extension. It is possible to load this screen into an art package for retouching. Note that the .GRA file is only 8000 bytes of bitmapped screen and that no colour information is saved. It is fairly simple to load this screen into any programs that you may be developing.

VIEW SCREEN
Allows you to view the current screen without disturbing it. It is useful to verify that everything is OK before printing or saving the screen. Press any key to return to the menu.

RETURN TO MAIN MENU
Simply returns to the main menu but note that the graphics screen is not disturbed until you select another graphing function so it is possible to execute DOS commands etc, and return to the graphics menu for saving the screen.

DISK OPERATIONS

LOAD STATISTICS
Allows you to recall statistics saved with the option below. Again, you must omit the .STAT extension from the filename as it will be automatically added. This can only load GRAPH-ED .STAT files and not SPREAD-ED .SHT files.

SAVE STATISTICS
Allows you to save statistics typed into GRAPH-ED in an easy and quick format for recalling using the option above. Statistics files are distinguished by a .STAT extension.

READ ERROR CHANNEL
Reads the disk drive error channel to determine what the cause of that annoying flashing LED is!

READ DISK DIRECTORY
Reads the disk directory of the disk in the drive. Press CTRL to exit.

DISK COMMANDS
Type in any normal DOS command and it will be sent to the device 8 drive. Press CTRL to exit.

RETURN TO MENU
This is obvious!!

RESTART PROGRAM
You will be asked to confirm your decision as all data will be cleared before restareting. The same effect can be achieved by holding down RUN/STOP and tapping RESTORE. You can only load spreadsheet files from the start of the program.

QUIT PROGRAM
Resets the computer back to Basic. Save all your files before exiting!

Thats just about it!! GRAPH-ED was designed to fulfill my need for graphs in my GCSE Geopgraphy project and people commented on how professional the graphs looked. I hope that you will have some use for GRAPH-ED and it may improve your work too!

RIPPING SOFTWARE - *is it legal?*

STUART ALLEN asks the age old question and comes up with his own ideas.

We've had an article about hacking, now for one about Ripping it. Ripping tapes is a bit difficult, because they are made of plastic, therefore harder to break. You could try ripping the piece of paper, which tells you about the software. Disks aren't that much of a problem, as they aren't as strong. Seriously, ripping software is taking out bits of code from a program, and using it in another. To make this a little bit easier, you should use a cartridge with an inbuilt M/C monitor. This would help you in two ways.:

1) You aren't using valuable memory of the computer, where that infamous small 10 bytes of code that you want lies., and
2) You can halt the programs execution (and no, that doesn't mean that you are killing it using a guillotine), and see what the interrupt is set to (either using M 0314 or M FFFE).

I put those two in, because at $0314, some programmers change the vector to point to a different location (like $4023, which gives them enough space to put in the interrupt loader, and to do a loop (JMP $4020). This would mean that at the start of every interrupt, it would check locations $0314-5 (normally set to $EA31) and do an indirect jump to the location that was held within. (i.e. JMP ($0314)) (You may have noticed by now that I like putting brackets everywhere. Think I'll use fancy ones next time) It also checks the location $FFFE (told you that I'd use fancy ones), although I've forgotten which one it checks first.

> **"TO RIP OR NOT TO RIP, THAT IS THE QUESTION."**

Another form of rippIng is changing things. This could be scrolltexts (even though I've seen a very strange way of storing the text. G in the memory would be F on screen, H would be G, etc) A special favourite is changing the picture (if there is one) or changing the colour bars, again, if there are any. Yet another thing that they sometimes do, is on demos where there are more than one part, they check for the link routine, and save the memory that goes with the next part.

The big question is... Is it illegal? Well, yes and no. The makers of cartridges, like the power cartridge or action replay, say this:

> **Power cartridge : Bitcon Devices Ltd does not authorize or purport to authorize the making by any means or for any purpose whatsoever of copies or adaptations of copyright works or other protected material, and users of the Power Cartridge must obtain the necessary prior consent for the making of such copies or adaptations from all copyright and other right owners concerned.**
> **Action Replay : WARNING**
> **1988 COPYRIGHT ACT**
> **Datel Electronics neither condones or authorizes the use of it's products for the reproduction of copyright material.**

The back-up facilities of this product are designed to reproduce only software such as Public Domain material, the users own programs or software where permission to make a back-up has been clearly given. It is illegal to make copies, even for your

own use, of copyright material, without the expressed permission of the copyright owner, or their licensee.

"WHETHER 'TIS NOBLER TO SUFFER THE COST OF ORIGINAL SOFTWARE ..."

Now what the games companies say. This is taken from the game ... TRAZ - TRanstormable Arcade Zone, made by CASCADE

All rights of the producer and of the owner of work(s) being produced are reserved by Cascade Games Ltd. Unauthorized copying, hiring, lending, public performance and broadcasting of this cassette/disk are highly prohibited The publisher assumes no responsibility for errors, nor liability for damage arising from it's use.

This is what they are saying, in English. Power Cartridge says 'if you have got any copyrighted software, then don't use this cartridge, unless you beg with the industries to let you copy stuff. 'Strictly speaking, you can't even freeze the game, because that would be altering it, in some way. Datel say, 'You can't copy copyrighted stuff, although you can alter it.', and Cascade say, 'Even if you use this piece of software a lot, you aren't allowed to copy it. It isn't our fault if you like the software so much...' There are ways, of course, of getting around the copyright law. If you use the M/C monitor in Datels Cartridge and change just one byte, then you will have altered the software from it's original form. Cascade own the copyright of the original software, but you own the copyright of your version because you have altered it. <What's all this got to do with ripping?> Well, once you have made that one byte alteration, you can rip to your hearts content. It is only illegal if you don't own the copyright, which once you have adjusted it slightly, makes it legal. At least, this is the theory of it all. Unfortunately, it doesn't quite work out that way. You can only rip things out of other peoples work (for games, that is) with their consent. On the other hand, demos aren't copyrighted unless they are demos of games by official companies. Aren't they rotten, eh? Just think, all those small little routines that you have wanted to do, that just made a tiny 'glitch' on the screen. You have tried for days on end in getting it off the screen, but to no avail. Then, comes the latest blockbuster game, with that one single routine that you were trying to code. With NO glitch.

Note to all non-demo peeps. A 'glitch' is where the raster line isn't perfectly straight.

End of note to all non-demo peeps. (and maybe to some of the small demo peeps!

What do you do????? Do you :

a) Stare at the screen, looking at it furiously?
b) Stare at the screen, asking it gently to 'Leap' out of the code and onto your disk or printer paper?
c) Rip the code out?

The obvious answer is part 'c'. It would be easier, and maybe there might be another small piece of code, that you have been meaning to code, and not had time. You never know until you try it. I am not proposing that you all rip hell out of the games that you get, but only if you have to. Another example: If, for some strange reason, your disk drive throws a wobblie and ruins a disk, with most of those long, hard to code, routines on. You know which programs have got in some code to do roughly the same job. The question is, what do you do?

a) Scream blue murder at the disk drive?
b) Scream blue murder at the disk?
c) Scream blue murder at the dog?
d) Rip like hell all those pieces of code?

I deliberately gave you more choices, to make it harder for you to find the right one. It is, obviously, part 'a'. Erm, sorry, I ment, part 'd'. You don't see why you should take more time and effort, in trying to remember what the pieces of code did, where they stood in memory, or how to code them, if you've got some similar routines in someone elses work. The only trouble is that you can also get called a LAMER if you make demos, with using solely other peoples routines. The only way around that, is to keep all the versions of the code, and not to credit anyone (apart from yourself). This should (will?) keep all the name-callers at bay. What they don't see, they can't criticize about. And what they can't criticize about, they can't call people names about. (Did you understand that? If you did, then please, tell me what I meant.) Sigh, the world is all too fierce for some folk. It's a shame, though. Why can't we all live in harmony? (and not slag other folk off about bits they ripped from other peoples work.

"... OR TO SUFFER THE CONSEQUENCES OF BEING FOUND OUT."

EDITORS COMMENT: The Editor of CDU and indeed the publishers do not condone any form of software piracy, no matter what the reasons. Software piracy will do for the computer world, what Video piracy has done for the film making world. However, the magazine will always allow the individuals to air their own views.

NUMBERS AND BYTES

A single part discussion on three numbering systems and their application to memory bytes.

JOHN SIMPSON

We all learnt from a very early age, how to count by using the common numbering system, DECIMAL. It has now become second nature to us, much like the English language. Some of us have a very good knowledge of the language whilst some of us do not. The same with numbers, some good, some not so. But, in any event, we all speak our tongue, and readily understand each other. The same can be said with numbers, we can all use them to good effect, if not brilliance. Not many of us have trouble with addition and subtraction when it comes to the receiving or giving of money!!

THE HEXADECIMAL SYSTEM

HEXADECIMAL, at first sight, and possibly second or third, may appear to be almost incomprehensible. (I know it did to me!) but with a little perseverance the blocks and the blinkers soon come tumbling down and clarity fills their place, and HEXADECIMAL becomes second nature, as did language and decimal. All right, let's just refresh our minds with a flashback to decimal.

0 through to 9, 10 through to 99, 100 through to 999 etc.

Remember how we used to evaluate the digits into the named columns?

1 3 5 2
- └ Units column
- Tens column
- Hundreds column
- Thousands column

and we can continue building up the columns - seemingly to infinity - tens of thousands, hundreds of thousands and so on etc

We also discovered in those early years that ten (10) was the base standard. What this meant was that each column would only add up in multiples of ten. 0,1,2,3,4,5,6,7,8,9, then cary over to the next column.

We were taught how to add together a couple of numbers and by using a carry, we could add the overflow from one column to the next. A simple example of this might be,

9 + 1 = 10

or

```
  9
+ 1
---
=10
```

Here the units column of the 9 is full, and so when we add the one to the nine, we reset the units column to a zero, and carry one to the tens column. Okay, then. Remembering this let us now have our first peek at HEXADECIMAL. All the rules of numbers still apply, and we still group the individual digits into columns. However, because the base number, 10 in decimal, has now been raised to sixteen (16), each column can now be added up in multiples of sixteen, or 0 (zero) through to fifteen. Fine, but we only have ten digits, so where are the other six?

Here are the first ten	And here are the other six
0 1 2 3 4 5 6 7 8 9	A B C D E F
0 1 2 3 4 5 6 7 8 9	10 11 12 13 14 15

Dealing with column names in decimal was simple because everything is powers of ten. Single units column times ten = tens column - times ten = hundreds column and so on. However, with HEXADECIMAL the powers are of sixteen, which means Single units column times sixteen = 256 - two hundred and fifty sixes column times sixteen = 4096 and so on.

An example of a HEXADECIMAL number is;

A2DF
- └ Units column
- Sixteens column
- Two hundred and fifty sixes column
- Four thousand and ninty sixes column

We could, the same as with the decimals column, continue adding in more columns, however, no point, because the largest number for a memory address you will require on our humble sixty-four, or any other 8 bit machine for that matter, can be contained within a four digit HEXADECIMAL number, which is; FFFF in decimal this is 65535.

You will quickly spot that much larger numbers can be held within the span of four digits in hex than can be held in a decimal four digit number. This is of great advantage when dealing with memory bytes, as we shall see

A simple example following in the footsteps of the earlier simple example is;

F + 1 = 10

or

```
  0F
+ 01
----
= 10
```

Here we see that the units column of the F is full so when we add the one to this we clear the units column by resetting it to zero, and carry one over to the sixteens column. Therefore when we find a 10 in HEXADECIMAL we know that this is the same as 16 in decimal. 11 in HEXADECIMAL is the same as 17 in decimal, and so on. If we continue to add one to the units column it would increase thus;

0+1=1+1=2+1=3+1=4+1=5+1=6+1=7+1=8+1=9+1=A+1=B+1=C+1=D+1=E+1=F

At this point when we add a further 1 we empty the units column to zero (0) and carry the one to the sixteens column (10). If we continue doing this eventually the sixteens column will also become F (F0). A further fifteen single increments to the units column and our HEXADECIMAL number now looks like this FF which is the same as decimal 255. You may have spotted that this value is the maximum value a memory byte can hold (255 decimal or FF HEXADECIMAL). Subtraction and multiplication follow the same rules of arithmetic as for decimal.

THE BINARY SYSTEM

In BINARY our base has now changed to two. This means that the maximum value which can be held in the units column is 1. Actually, the units column changes it's name in BINARY. It is now refered to as the ones column. The columns in BINARY can hold the value of either 0 or 1. The same rule applies as you add values to a column, namely, as the column reaches the base value, then the column reverts to a zero, and a carry is taken to the next column to the left. So, we know that the only digits in the BINARY numbering system are a 1 and a 0, and that the base is 2. Therefore the next column after the ones column is the one times base (2) = twos column, two times base is fours column, and fours column times base is the eights column, and so on. Let's look at a four digit BINARY number;

```
1 0 1 0
  | |
  | └── Ones column
  └──── Twos column
        Fours column
        Eights column
```

A further example following in the footst....(okay, we've got it. . .ED!).

1 + 1 = 10

or

```
  1
+ 1
---
=10
```

In this example we can see that the ones column of the first digit is full, it has reached the maximum value. So when we add a further 1 to it the column resets to 0 and we carry one over into the twos column. If we now add another 1 to the ones column the BINARY number will change from this, (10) into this, (11). If we didn't know we were dealing with BINARY we might mistake the number for eleven. But let's now add a further one to the ones column. Well, it can't become 12, so reset the ones column to 0 and carry one to the twos column and add it to whatever is already there. In our case, another 1. It can't become 20, so set the twos column to 0 and carry the one to the fours column. Now we should have a number which looks like this; 100. Once again if we did not know our base was two, and thought we were observing a decimal number, we would conclude that the value was one hundred, and not the four (4) which it actually is.

Right then, each time we add one to a BINARY number which already equals one we do a carry to the next column. Unlike HEXADECIMAL which could hold a higher value in four digits compared with decimal, a four digit BINARY number is far less.

BINARY	DECIMAL	HEX
0000	0	00
0001	1	01
0010	2	02
0011	3	03
0100	4	04
0101	5	05
0110	6	06
0111	7	07
1000	8	08
1001	9	09
1010	10	0A
1011	11	0B
1100	12	0C
1101	13	0D
1110	14	0E
1111	15	0F

RECOGNITION OF BASE

In order to let us know what number base we may be looking at when we observe a number, the symbols $ (dollar sign), and the % (percent sign) are used to pretix hexadecimal and binary numbers and no sign at all represents decimal.

1000 - Decimal
$1000 - Hexadecimal
%1000 - Binary

DEC, HEX, and BIN for BITS, BYTES 'N' NYBLES

There are 65535 memory cells, or locations contained in the Commodore 64. These are also refered to as bytes. Each byte is an eight-bit BINARY number, with the minmum value of 0 (zero) to the maximum value of 255 ($FF, %11111111). Let's examine a byte a little further I as in the table below.)

To evaluate the byte we add together the column values of the bits set to 1. For example;

```
Byte = 10011001
          └──   1
       +  8
       + 16
       +128
       ----
       =153  bit value
             in decimal.
```

7 6 5 4 3 2 1 0 bit number					
Bit No.	Bit Value		Columns	Expressions	
	Dec	Hex		Pairs	Nybles
0	1	01	1's column	bit	
1	2	02	2's column	pair	low
2	4	04	4's column	bit	nyble
3	B	08	8's column	pair	
4	16	10	16's column	bit	
5	32	20	32's column	pair	high
6	64	40	64's column	bit	nyble
7	12B	80	128's column	pair	

However, using HEXADECIMAL this becomes even more simple. We start by dividing the byte into half. The lower four bits constitute the low nyble and the upper four bits the high nyble. Each nyble can now only hold a maximum value of $E0 (high nyble) and $0F (low nyble). So to evaluate a byte with a value of 11101011 using HEX we;

a) Split the byte into two nybles

HIGH	LOW
1110	1011

b) Add together High nyble 1 bits

1110

$2
+ $4
+ $8
———
+ $E

c) Add together Low nyble 1 bits

1011

$1
+ $2
+ $8
———
= $B

d) Recombine the hex nybles

High	Low
$E	$B

The value of the byte becomes $EB

Because a byte can only hold the maximum value of $FF (255), then to address bytes which are higher in memory than $EE we must combine two bytes together to form a 16-bit word. Each byte of the 'word' is said to be the high byte and the low byte. For example;

11111111	11111111
High byte	Low byte

The first column on the right of the high byte becomes the 256's column, the next becomes the 512's column and so on. Now, if you care to calculate, we can hold a value ranging from 0 (zero) to 65535 - which is the total number of bytes in the C64. We can address any memory location by using a high/low byte address pointer.

To find the address a word holds, just divide into four nybles and add together, in HEX, the set bits in each nyble, then recombine giving you a four digit HEX value.

BITS USED AS SWITCHES OR FLAGS

If you have a byte of data where each bit may represent a flag, or a switch, either true, false, on or off, of some encoded unit or item - such as the eight sprites - then the bitwise logical and/or statements are the tools for setting and resetting the bit (switch/flag).

Let's take an example using the sprites. You have a situation whereby you are displaying a combination of the sprites on the screen, sometimes they need to be on (enabled) and sometimes off (disabled).

1. To turn on (set) a bit.

Sprite bit numbers	7 6 5 4 3 2 1 0
Present sprites on	1 0 0 1 0 0 1 1

You decide to turn on sprite 5.

Using a temporary byte create a mask with all the bits reset to zeros, except those bits you wish to turn on - in this example bit 5.

Bit numbers	7 6 5 4 3 2 1 0
Mask	0 0 1 0 0 0 0 0

Now place the mask beneath the sprite bits.

Bit numbers	7 6 5 4 3 2 1 0
Sprites	1 0 0 1 0 0 1 1
Mask	0 0 1 0 0 0 0 0
Result of logical OR	1 0 1 1 0 0 1 1

When we OR the two sprites bytes and mask with each other, if either corresponding bit is a 1 then the corresponding bit in the result is a set '1' bit. If both bits are reset 0, then the corresponding result bit is also reset to a 0. When all the bits are evaluated the result is placed into the sprites byte (more commonly refered to as the sprite enable register). This, then, updates the byte, or register, and switches on (enables) the sprite.

To use this from Basic a command such as the following could be used;

```
10  POKE SPRITE,PEEK(SPRITE)
    ORSPRITENUMBER
```

Where sprite=53269 (the byte in the VIC chip which controls the sprite enable/disable), and SPRITENUMBER = 0 to 7.

From Assembler you could use something like;

```
LDA SPRITE
ORA SPRITENUMBER
STA SPRITE
```

2. To turn off (reset) a bit.

We decide to turn off sprite 5. This time we create the mask byte and set all the bits to 1 except those bits we wish to turn off.

Bit numbers	7 6 5 4 3 2 1 0
Mask	1 1 0 1 1 1 1 1

and again compare it with the sprites byte;

Bit numbers	7 6 5 4 3 2 1 0
Sprites	1 0 1 1 0 0 1 1
Mask	1 1 0 1 1 1 1 1
Result of logical AND	1 0 0 1 0 0 1 1

When we AND the two sprites bytes and mask with each other, it both bits are either 0, or one of the two bits is 0, and the other 1 then the result is a 0. If both bits are set 1, then the result is a 1.

To use this from Basic, a line such as the following could be used.

```
10  POKE SPRITE,PEEK(SPRITE)
    AND(255-SPRITENUMBER)
```

and from Assembler

```
LDA SPRITE
AND #255-SPRITENUMBER
STA SPRITE
```

Well, that concludes this very short discussion for you beginners. I hope it has helped somewhat for those of you that desire to understand a little more about Hexadecimal, Binary and Bytes.

BACKGAMMON

Play a game or two of this excellent Backgammon simulator and become a champion

PETER WEIGHILL

Backgammon is a board game in which the aim is to move all fifteen of your counters around the board, and then off, before your opponent. If you have not played the game before and do not know the rules then you should read the instructions included in the game

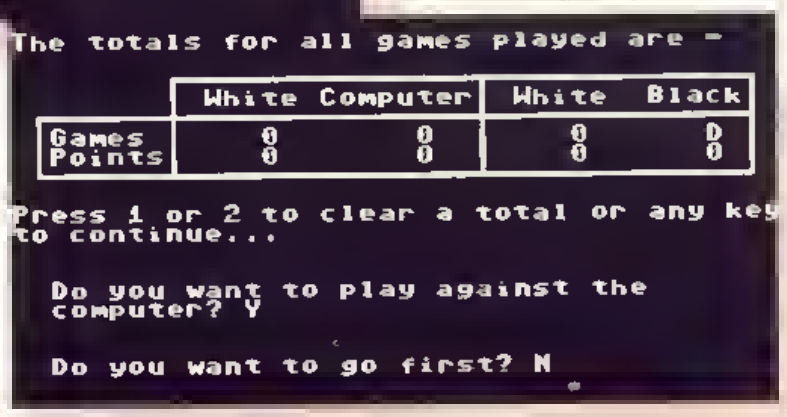
After the instructions, the scores are displayed from previous games These will be all zero on your first go. If you want to clear the scores press 1 to clear the one player game scores and 2 for the two player game.

You should then press any key to continue.

You will then be asked whether you want to play the computer. Type in Y or N and press return.

If you typed in Y then you will be asked whether you want to go first. Type in Y or N and press return.

Now the board will appear and the game will begin.



HOW TO ENTER A MOVE

You will be asked the question 'Which piece to move?'. You should type in the number corresponding to the piece you want to move (the numbers are displayed around the board)

If one of your counters has been knocked off the board by your opponent then you should type O to the above question to get your counter back on the board. You must re-enter this counter before you can move any other.

If you cannot move then type N to the above question.

Once you have typed in the piece to move then you will be asked 'Which dice to use?'. You should type in the value of one of the dice displayed in the top right corner and press return

If you have typed in the wrong counter to move and only notice the mistake when you have to input a dice value then type 0 and return.

PLAYING THE COMPUTER

The computer is always White It will display its moves in a column on the right. After it has finished moving its pieces then you should press any key to continue. You should then input your moves

AT THE END OF THE GAME

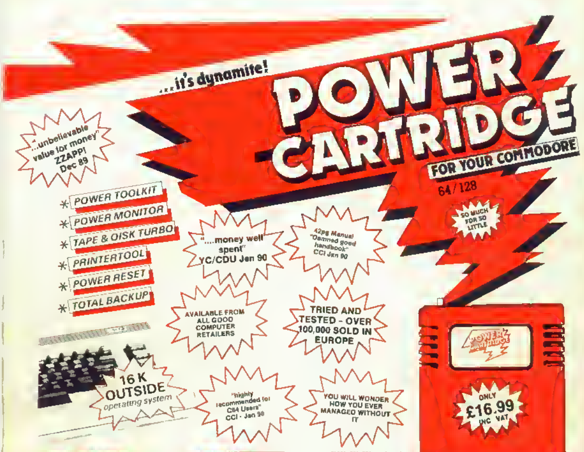
The game finishes when one player has taken all fifteen of his counters off the board. The points won will then be displayed and added to the totals for all games played.

HOW POINTS ARE SCORED

The player who wins the game receives one point for each opponents counter left on the board.

The player will receive double points if the opponent still has all fifteen counters on the board.

The player will receive triple points if the opponent also has one or more of his counters in the winning players home table.



A powerful BASIC toolkit (Additional helpful commands) that considerably simplifies programming and debugging.

AUTO	HARDCAT	RENUMBER
AUDIO	HARDCOPY	REPLAE
COLOR	HEX$	SAFE
DEEK	INFO	TRACE
DELETE	KEY	UNNEW
DOKE	PAUSE	QUIT
DUMP	PLIST	MONITOR
FIND	PLOAD	BLOAD

RENUMBER: Also modifies all the GOTO's GOSUB's etc. Allows part of a program to be renumbered or displaced.
PSET: Set up of printer type
HARDCAT: Prints out Directory

The toolkit commands can be used in your programs.

Using POWER CARTRIDGE you can load up to 6 times faster from disk.
The Disk commands can be used in your own programs.

DLOAD	DVERIFY	DIR
DSAVE	MERGE	DEVICE
DISK		

MERGE: Two BASIC programs can be merged into one.
DISK: With DISK you can send commands directly to your disk.

Using POWER CARTRIDGE you can work up to 10 times faster with your data recorder. The Tape commands can be used in your own programs.

LOAD	SAVE	VERIFY
MERGE	AUDIO	

A powerful machine language monitor that is readily available and leaves all of your Commodore memory available for programming.
Also works in BASIC-ROM, KERNAL and I/O areas.

A ASSEMBLE	I INTERPRET	S SAVE
C COMPARE	J JUMP	T TRANSFER
D DIS-ASSEMBLE	L LOAD	V VERIFY
F FILL	M MEMORY	W WALK
G GO	P PRINT	X EXIT
H HUNT	R REGISTER	$ DIRECTORY
		DOS Commands

The POWER CARTRIDGE contains a very effective Printer-Interface that self detects if a printer is connected to the Serial Bus or User Port.
It will print all Commodore characters on Epson and compatible printers.
The printer-interface has a variety of set-up possibilities. It can produce HARDCOPY of screens not only on Serial printers (MPS801, 802, 803 etc) but also on Centronic printers (EPSON, STAR, CITIZEN, PANASONIC, etc).
The HARDCOPY function automatically distinguishes between HIRES and LORES. Multi-colour graphics are converted into shades of grey. The PSET functions allow you to decide on Large/Small and Normal/Inverse printing.
The printer PSET functions are:

PSET 0	Self detection Serial/Centronics
PSET 1	EPSON mode only
PSET 2	SMITH-CORONA mode only
PSET 3	Turns the printing 90 degrees!!
PSET 4	HARDCOPY setting for MPS802/1526
PSET B	Bit-image mode
PSET C	Setting Lower/Upper case and sending Control Codes
PSET T	All characters are printed in an unmodified state
PSET U	Runs a Serial printer and leaves the User port available
PSET Sx	Sets the Secondary address for HARDCOPY with Serial Bus
PSET L1	Adds a line-feed, CHR$ (10) after every line
PSET L0	Switches PSET L1 off

On the back of the POWER CARTRIDGE there is a Reset Button. Pressing this button makes a SPECIAL MENU appear on the screen.
This function will work with many programmes.

CONTINUE	Allows you to return to your program
BASIC	Return to BASIC
RESET	Normal RESET
TOTAL BACKUP DISK	Saves the contents of the memory onto a Disk. The program can be reloaded later with BLOAD followed by CONTINUE
RESET ALL	RESET of any program
TOTAL BACKUP TAPE	As BACKUP DISK but to TAPE
HARDCOPY	At any moment, prints out a Hardcopy of the screen. Using CONTINUE afterwards you can return to the program
MONITOR	Takes you into the Machine language Monitor

BDL

Bitcon Devices Ltd

88 BEWICK ROAD
GATESHEAD
TYNE AND WEAR
NE8 1RS
ENGLAND

Tel: 091 490 1975 and 490 1919 Fax 091 490 1918
To order: Access/Visa welcome - Cheques or P/O payable to BDL
Price: £16.99 incl. VAT.
UK orders add £1.20 post/pack total - £18.19 incl. VAT.
Europe orders add £2.50. Overseas add £3.50
Scandinavian Mail Order and Trade enquiries to: Bhiab Elektronik, Box 216, Norrtalje 76123, SWEDEN Tel: + 46 176 18425 Fax: 176 18401
TRADE AND EXPORT ENQUIRIES WELCOME